Parting Thoughts

Steven M. Bellovin

CS@CU

Structure of This Talk

- The boring stuff:
  - How I got started in computing
  - What I did in grad school, at Bell Labs, etc
  - What I've done at Columbia
- Where I think security is going (and should be going)
- Where I think tech policy is going (and should be going)
- Parting thoughts

CS@CU

The Early Years

I’m Retiring

- I’m no longer teaching
- As of June 30, I won’t have an office (though I’ll be a regular faculty member through June 2025)
- Not sure how much longer I’ll be living in Manhattan

The Early Years

- I learned to program at age 14, when that was very unusual; my high school (Stuyvesant, here in NYC) got a computer (an IBM 1130) the following year
- The students who knew programming ran it by ourselves
  - During my senior year in high school, I wrote a disassembler to help me understand the operating system's internals

More details at https://www.usenix.org/system/files/login/articles/07_bellovin.pdf

College

- I was an undergrad at Columbia College at a time when there was no CS major
- I made up my own major—effectively CS—and had it approved by the Committee on Instruction
- I worked as a systems administrator/systems programmer throughout college, first at Teacher's College, then IBM, then two years at CCNY, the computing hub for all of CUNY
- I also hung out a lot at the Columbia University Computer Center
- There and at CCNY, I started thinking about computer security

CUCC

- The university had a 360/91, a supercomputer (by the standards of the day)
- It had 2MB of RAM—huge!
- This picture is of just the console!
- Most programs were submitted via punch cards
- There were a number of IBM 2260 CRT terminals attached—which brought some security issues…

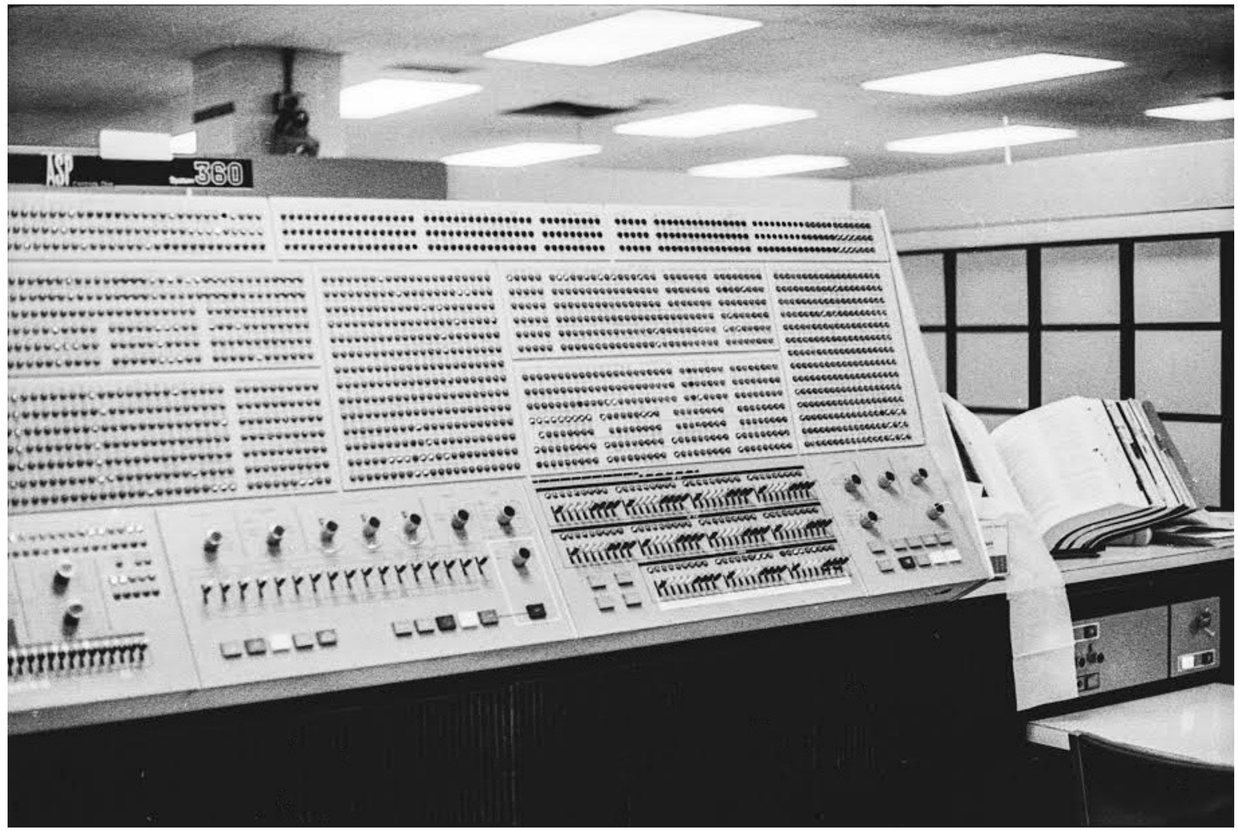


Photo by me, Spring 1972

CS@CU

The IBM 2260

- The 2260s were used for online text editing (of punch card images) and job submission
- One hack: a fake login screen, to capture passwords
- Another: when you hit ENTER, the system read from the “Start Symbol” to the “End of Line Symbol” inserted by the ENTER key
- Display a line with a start symbol, a nasty command, and an end symbol—if that was earlier on the screen than the real ones, the nasty command would be read…

Photo from https://en.wikipedia.org/wiki/IBM_2260

CCNY

- Use for administrative and academic computing
- All program submission was on punch cards
- Disk storage was limited; removable disk packs were the norm
- A computer operator noticed an unexpected disk mount request and ask me about it
- I looked at the punch cards, from two students. One I hired, the other I sent to the dean…

https://en.wikipedia.org/wiki/IBM_2314#IBM_2314_Disk_Access_Storage_Facility_Model_1

Other Hacks and Stunts at Columbia

- I had a number of similar minded friends
- On April 1, change all of the messages for the online system
  - “LOGIN:” ➔ “WHO’S THERE?”; “PASSWORD:” ➔ “PROVE IT!” by changing the command deck for system startup to bind to a different file, and to change the module search path
  - We never modified any code
- Fed object files to the compilers to see what they’d do—it wasn’t pretty…
- Did you know that you could “dial” a number on a locked rotary dial phone by bouncing the hook switch with the right timing and rhythm? (In those days, all modems had actual telephones as a required component.)
- Thought experiments on what would happen if we had a self-replicating program that used the “internal reader” to submit two copies of itself—but we knew better than to try it…

What Computer Stuff Did I Learn at Columbia?

- By the time I graduated, I knew many languages: FORTRAN (II and IV), PL/I, APL, BASIC, ALGOL 60, SNOBOL 4, IBM 1130 and IBM 360 assembler (and the macro language for S/360 assembler, itself a powerful language), PL360
  - None of these are particularly useful today…
- Data Structures was a 6000 course—and the prof, who knew me well and knew I was qualified, didn't want me in the class but couldn't keep me out, because I was in the College…
- A lot of very useful experience with real, high-end operating systems
  - Earlier, I'd learned how virtual memory and timesharing work by reading Butler Lampson's lecture on the SDS 940

Grad School: U. of North Carolina at Chapel Hill

- A small department, founded and chaired by Fred Brooks
  - Brooks had managed the development of the IBM S/360 mainframes, a tremendous success
  - He'd also managed the development of OS/360, which he regarded as a failure. Why?
  - I took four courses from Brooks, including software engineering and computer architecture—he was a tremendous influence on my professional career and outlook

What Did I Learn in Grad School?

- I learned that there were useful formalisms
  - For example, I'd never heard of finite state automata
- I learned how to *think* like a computer scientist
- A few more languages (Algol W, Algol 68, PDP-11 assembler, PL11, Pascal—and C, plus the earliest versions of C++)
  - Guess which are still useful…?
- I didn't think I was doing security, but my dissertation, on formal verification of the output of a compilation, is in fact security-relevant (yes, I did a theory dissertation)
  - It would have caught Thompson's compiler-resident back door
- I was still doing system administration

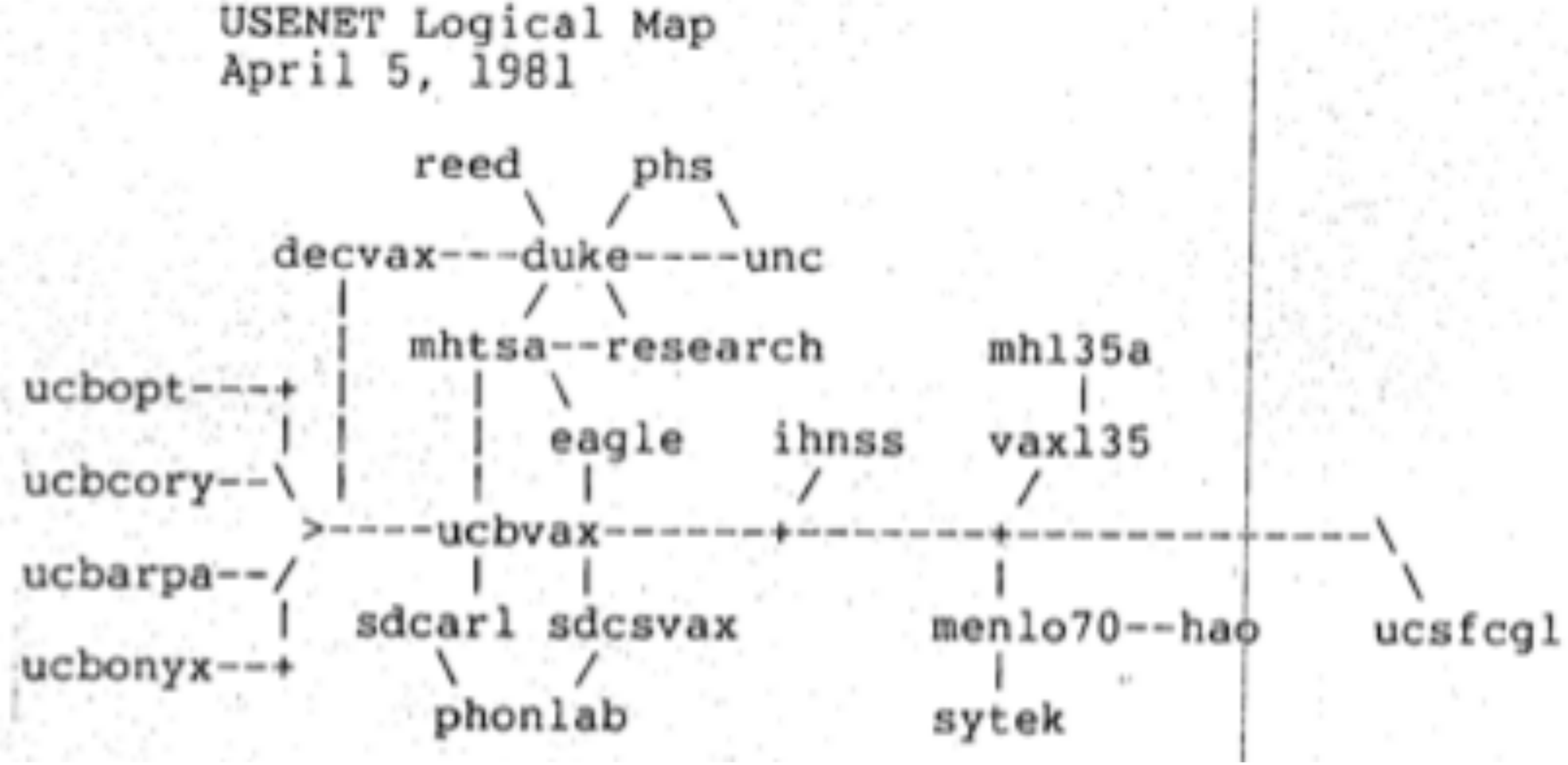
Security

- I read the Morris and Thompson paper on passwords, and implemented it
  - One friend, when I told him that his password was “abscissa”, replied “no one else can spell it”
- I had the dubious privilege of suspending programs being run by a full professor who didn’t understand the difference between real memory and virtual memory, and who ran enough instances at once to drive the machine to thrashing
  - Brooks: “If someone is entrusted with the root password, they have not just the right but the responsibility to use it when necessary.”
- Usenet…

Netnews (AKA Usenet)

- I helped create Usenet—but we knew that it wasn't managed and we knew that that might be a problem
- How to authenticate users in a very large, distributed network?
- We knew about public key cryptography but didn't know how to *engineer* a solution
  - We didn't know of the invention of certificates, which we thought would have solved our problem—but they wouldn't have solved it without a PKI
  - It's just as well—we also didn't know about the export laws around cryptographic code…
- And then I joined Bell Labs

Usenet Map, April 5, 1981



Caption

The Labs

Why I Joined Bell Labs

- I really wanted to become a professor—I loved teaching, and in fact had seven semesters of teaching experience, over four different courses, by the time I graduated
- But—I'd seen what lack of money could do to cripple research (and make life awful for students)
- I suspected (correctly) that I'd hate anything and everything to do with grants
- I wanted to go someplace where I'd have the resources to do the research I wanted to do, without scrounging; at the time, Bell Labs was that place

Bell Labs

- Primary early accomplishment: helping bring TCP/IP to Bell Labs
- I “owned” 1½ of the first three Ethernet networks in all of AT&T, a company with 1 million employees when I joined (though that was about to drop sharply because of a court-ordered breakup)
  - I’d been doing communications since college: remote job entry at IBM and CCNY, linking our Unix machines in grad school, writing device drivers, etc.
  - My first published paper was on routing in a dial-up network
- There were outages due to configuration errors on the Bell Labs TCP/IP network—and I wondered what would happen if someone did those things deliberately…
- I also started learning cryptography

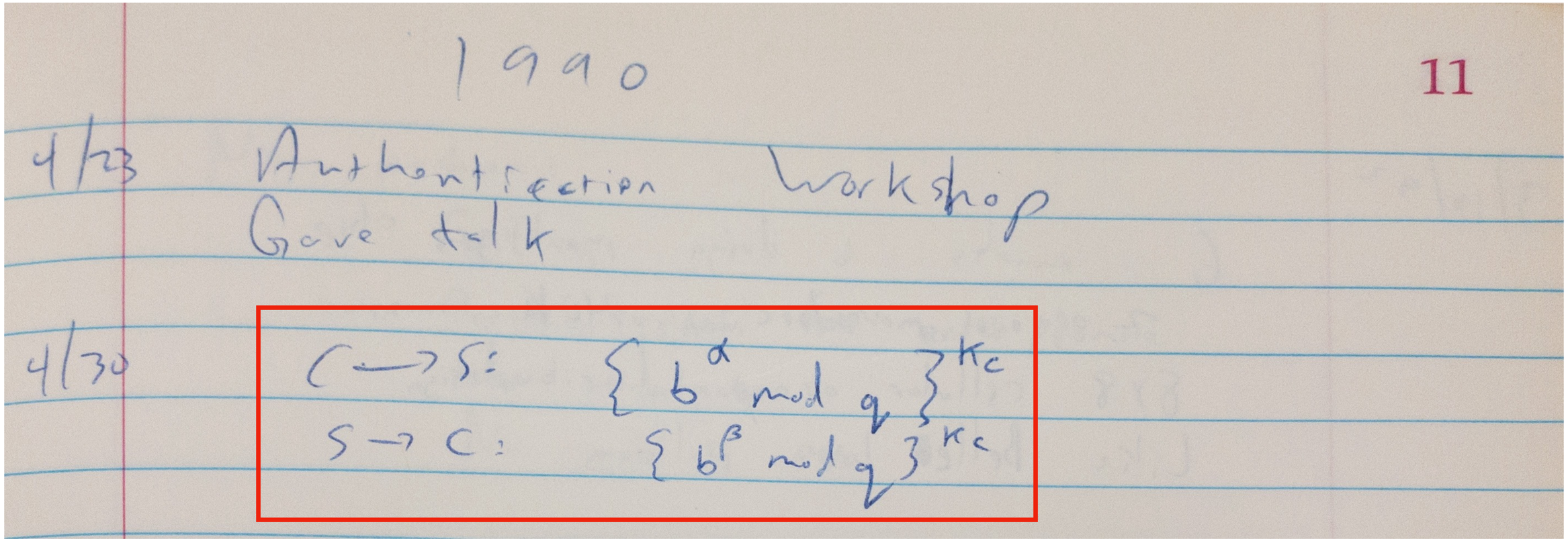
Internet Security

- Around 1986, I started reading the protocol RFCs from a security perspective: were there design flaws in the protocols?
- Yes, per my first major paper, in 1989
  - Major issues: routing security and sequence number attacks
- I started thinking about packet filters as a firewall, and set up an experimental link to another company to test this
  - I got scooped by others
- I worked on connecting AT&T to the early *Internet* (not *ARPANET*)

Password Guessing

- I'd of course known about password guessing attacks ever since the 1979 Morris and Thompson paper
- That required a (hashed) password file
- But—Kerberos used a password to protect the initial cryptographic exchange—could an attacker run password guessing on such packets
- Sure!
- Was there a defense? I thought about it for several months…

EKE: Encrypted Key Exchange



The most original idea I've ever had, which came to me during a very boring talk…

A Fateful Train Ride…

- On the train to Baltimore for Usenix Security in 1993, I ended up in the same car as Bill Cheswick
- We started talking about a book on Internet security—there were none at the time—and we planned a collection-of-papers book
- Soon after the conference, John Wait, an editor at Addison-Wesley, stopped by for his usual annual visit: After some chit-chat: “Do you want to write a book, Steve?”
- “Well, yes—here’s the table of contents.”
- “Collection-of-papers books don’t sell well—I bet you can do a real book.”
- The result was *Firewalls and Internet Security: Repelling the Wily Hacker*. The attention it got probably made Bill’s and my careers.

CS@CU

Were Firewalls the Answer?

- Firewalls were the correct answer *in 1994*
- They were the only available *scalable* defense mechanism, including against buggy code, and there weren't many protocols of interest that they'd have to deal with
- From a talk, in 1994, a month after the book came out:
  - "Network security is not the problem.
  - "Firewalls are *not* a solution to network problems. They are a network response to a host security problem.
  - "More precisely, they are a response to the dismal state of software engineering; taken as a whole, the profession does not know how to produce software that is secure, correct, and easy to administer. Consequently, better network protocols will not obviate the need for firewalls. The best cryptography in the world will not guard against buggy code."

What Should Replace Firewalls?

- Modern policies are *extremely* complex, with very many protocols and endpoints, and very rich external connectivity
- Distributed firewalls?
- Better security architectures
  - A concept I've been grappling towards since 1994
  - Basic concepts: strong isolation between functions; carefully defined and controlled interfaces between them
    - Example: Web servers should not consult an authentication database; rather, they should consult an authentication server via a *simple* protocol
  - With luck, the subject of my next book, *Designing Security*

Public Policy

- Circa 1993, three law and policy issues came up that AT&T was very interested in
  - CALEA—standardized wiretap interface for phone switches
  - The Digital Millennium Copyright Act (DMCA)
  - Backdoored cryptography, i.e., the Clipper chip
- Matt Blaze and I were the only people in Bell Labs Research who a) understood the technology; and b) were willing to talk to lawyers
  - The *Firewalls* book had a legal chapter—writing it was fun
  - Matt and I both teach law courses now…
- The trick: getting AT&T to want what we wanted

The Internet Engineering Task Force (IETF)

The IPng Directorate

- By 1993, it was clear that the Internet was going to run out of IP addresses
- Address translation looked infeasible—which was wrong, but we didn't know that—and would have serious disadvantages (which was both right and wrong)
- The IETF set up a directorate to pick a replacement; I was asked to join it
- The result was IPv6; I was one of the people responsible for IPv6 addresses being 128 bits
  - I still regard that address size as the correct decision
- We underestimated the difficulty of transition…

IPsec

- My main technical interest in the IETF was IPsec: provide ubiquitous, transparent security for all applications, at the network layer
- During the discussions, I led the fight to remove sequence numbers from IPsec, since the IP service model allows for duplicated, reordered, etc., packets
- I later realized that that was *wrong* (in one of my favorite papers), and led the fight to put sequence numbers back in—and add mandatory integrity protection as well
- Working on IPsec led to another favorite paper, on probable plaintext cryptanalysis: how do you know if you've found the key if you don't have a "crib"?

The Internet Architecture Board (1996-2002)

- I was named to the IAB in 1996
- The IAB is the overall oversight body for the IETF
- It looks at higher-level architectural decisions
- It's also responsible for external relationships and liaisons—more public policy…

The Internet Engineering Steering Group (2002-2004)

- The IESG manages the protocol definition process
- I became Security Area co-director in 2002—and as such, partially responsible for the security of *all* IETF protocol designs
- Major accomplishment: making sure that all protocols had *real* security analyses—no more “Security issues are not discussed in this memo” in RFCs
- I also co-chaired the Intellectual Property Rights working group—more law and policy…
- I had to step down when I joined the Columbia faculty—being an area director is a near-full time job

CS@CU

Columbia

Transitions

- It was time to leave AT&T Labs—Research
- AT&T was in financial trouble; the phrase “death spiral” was bandied around in the Labs
- Research management had changed—the head of research was no longer a very senior technical person, it became harder to publish, difficult to get travel money, and almost impossible to talk to the press
- Besides, I wanted to do something different
- I decided to do what I’d long wanted to do: become a faculty member
- Columbia was my undergraduate alma mater

My Columbia Experience

- I *love* working with students
- I love teaching
- Unexpected benefit: the Columbia Library Network—I have many odd interests
- Dealing with grants was even worse than I had thought…
- Academic freedom…

Academic Freedom

- It's wonderful no longer having to have a chaperone when I talk to a reporter
- It's even better not being accompanied by a political commissar when I venture inside the Beltway
- I can write papers on oddball subjects—cryptologic history, law, etc.—without worrying about what my manager will think
- But—there's only so much of that I can do without grants, and it's hard to get grants for that sort of thing
  - Everyone talks up interdisciplinary work, but few want to support it
  - A major challenge for academe!
- Government work

Government Work

- I was on two Homeland Security and one Election Assistance Commission advisory committees
- I spent a year working full time at the FTC
  - I could not have done that while at AT&T; it would have been a serious conflict of interest
- A few years later, I worked part-time at the Privacy and Civil Liberties Oversight Board—and there might have been a conflict of interest with AT&T
- These jobs were very rewarding—I accomplished things that affect many people, even if I can't talk about them
- Note: the Dean and the Provost were very supportive of those latter jobs

A Turn Towards Law and Policy

- In the last 10 years, more and more of my attention has been focused on law and policy
  - Again, these have long been an interest, but they're now a major professional focus
  - This has affected my teaching (though the *Anonymity and Privacy* seminar was the first course I taught when I came to Columbia)
- Why?

Computers and Society

- I also created COMS W3410, *Computers and Society*
- Many Columbia undergraduates care about the connection
- Worth noting: my interdisciplinary courses have had a more diverse student body than my straight CS courses. I decline to speculate on why.

Why Law and Policy?

Why Law and Policy?

- Because it’s interesting and fun?

Why Law and Policy?

- Because it's interesting and fun?
- More substantively: what we do has to fit into a larger societal context
  - Can I work on strong cryptography if back doors are mandated by law? (Backdoored cryptography is *not* strong)
  - The threats to our privacy are increasing
  - In places with strong privacy laws, computer systems must comply. How?
  - Technology is eating the world, but lawyers, judges, and legislators often don't understand it correctly, including its limitations
- I also started losing interest in pure systems security work—and instead of learning ML, I learned law…

CS @CU

Whither Systems Security?

What is “Systems Security”?

- NIST: “The protection of information systems against unauthorized access to or modification of information, whether in storage, processing or transit, and against the denial of service to authorized users, including those measures necessary to detect, document, and counter such threats.”
- Technopedia: “Information systems security, more commonly referred to as INFOSEC, refers to the processes and methodologies involved with keeping information confidential, available, and assuring its integrity.”
- Me: Practical measure for assuring the security of “systems”, ranging from single applications to large networks of computers dedicated to a particular function.

A Reactive Discipline

- Traditionally, systems security has been a *reactive* discipline—we find was to combat new threats
- Examples:
  - Stack canaries came after stack-smashing attacks—so attackers launched heap overflow attacks
  - W^X combats code injection—so attackers launched code reuse attacks, e.g., ROP

The Field Was Once Broad and Powerful

- We've had firewalls for ~35 years
  - They're a scalable policy enforcement tool, but connectivity needs are far greater today, and the protocols more complex, than in the early 1990s
- We now have powerful encryption tools to protect data, especially data in motion
  - We even have a PKI, though most users have no idea what it is, what it does, or what the errors it generates mean
- Stack buffer overflows were once ~50% of all attacks; today, they're all but gone
- We know how to do secure 2FA and how to prevent SQL injection attacks
  - Usability—human factors—and programmer training remain issues

But Where is the Field Going?

- And why?

Threat Models

One approach is to expand our threat models

- Teenage joy hackers?
- Folks after money?
- Current or former insiders?
- Intelligence agencies?
- But that can be a rabbit hole, especially if you start thinking about supply chains
  - (Who made the memory chips on your laptop?)

The Field Has Narrowed!

- Attacks have gotten more complex—vulnerabilities are now chained together
- We're chasing each specific problem—but why?
- We're still reactive—but each reaction protects against fewer and fewer attacks

Walls and Doors

- Security has long been a matter of “walls” and “doors”
  - The walls separate different contexts; the doors allow for *controlled* flow of information between the contexts
- Old example: kernel versus user space, where system calls are the doors
- Firewalls separate the enterprise from the Internet
- Virtual machines are separated from the hypervisor and each other, but can make hypercalls and talk over pseudo-networks
  - These communication paths are doors, and can and do have weaknesses

Walls

- We used to be pretty good at building walls
- Of late, they've proved leakier than one would like
  - Example: side-channel attacks
- That's partly because we're building more complex systems, and partly because attackers have upped their game

Doors

- We're *lousy* at doors
- Sometimes, the problem is the implementation
- More often, the policy—what we let through the door—is wrong
- That's often a function of the complexity of our system—we have to create complex policies to get anything done, but (as with anything else) complexity is the enemy of correctness, and hence in this case of security

Where Do We Go?

- Reactivity isn't a great strategy—the stakes are ever higher, and we may not have time to react
  - Change Healthcare "warned that, financially, the total cost of the cyberattack is estimated to be between $1.35 billion and $1.6 billion for calendar year 2024."
  - And then there's critical infrastructure
- Worse, each new defense protects against less and less
- We don't have a theory of what defenses to pursue

All That Said

- We still need reactive defenses
- Old vulnerabilities don’t go away by themselves
- Even if a weakness is presumed “unreachable”, modern exploits chain vulnerabilities together
- “‘If you live, remember the sanitation crew. You need us.’” (Jerry Pournelle, *The Prince)*

Another Approach: Valar Morghulis

- Assume that *any* system component can be compromised
- Now what?
- Zero trust is one approach, but it isn't quite right

Another Approach: Valar Morghulis*

- Assume that *any* system component can be compromised
- Now what?
- One answer: intrusion detection
  - OK, but then what? Damage is often O(1)
- How do we build *resilient systems?*

*“All men must die”, from George R.R. Martin’s *Song of Ice and Fire* saga.

Resilient Systems

- A resilient system fails gradually
  - Example: encrypt credit card numbers with user passwords. The rate of card number compromise is now the rate of password guessing (or perhaps site login activity).
  - Example: Apple's new theft protection feature on iPhones
- But what is a resilient failure mode for, say, a power grid controller?

Modules, Computers, and Interconnections

- Often, part of the answer will be how we divide up functionality
- In other words, stronger walls and simpler policies controlling doors
- But we need clear design principles, programmer and architect training, and tools
  - And all of that is, of course, threat model-specific
- None of that exists today

Usability as a Defense

- Many failures are due to failure to take into account how people behave
  - Make that **MANY** failures, including phishing, password rules, ACL configurations, and many more…
  - Example: we know that passwords are a failure. But MFA is resisted, and Passkey hasn't gotten much traction yet. Why? What can we do?
- There have been no new releases of *H. sapiens* lately, nor even any patches
- Computers should be built for the people, not people for computers

Usability is Hard

- People are non-deterministic
- Failure modes are very hard to predict
- It’s difficult and expensive to test human-centric solutions
  - Enough subjects
  - Proper controls
  - Social factors
  - Etc

Morningside Park, March 2023

Usable Systems are Important

- A large class of attacks succeed because of usability failures
- Stolen credential attacks—a major vector today—succeed because the available solutions are not user-friendly
- (We also know that yelling at users or calling them “lusers” doesn’t work very well…)

Machine Learning and Security

- Except for intrusion detection, machine learning is a comparatively new branch of security
  - (Probably because it's only in recent years that ML has started working well anywhere…)
- What else can it do? Code analysis? Code generation?

Training Data

- ML systems depend on training data
- There are many examples of bad code out there, and few examples of *known* secure code
- Generative AI can echo the bad code as easily as the good
  - Informal experiment with ChatGPT: two buggy programs and one insecure one
  - (ChatGPT seemed to have been trained by Little Bobby Tables' mom)

Other Issues

- Unpredictable behavior—no one really knows why complex ML systems create the output that they do
- Adversarial ML—can attackers drive your system into weird states?
  - Little Barbie Images?
- Model stealing—are we creating a new target for attackers?

Whither Systems Security?

- We need a better approach to the problem
- Reaction is not sufficient, though a theory behind it will help a lot
- My approach is resilient system design
- At least two possible broad spectrum defenses, if we can make them work: usability and ML
- Will more secure hardware help? What policies should it enforce?
- Others?
- *If we don't figure out something, I fear that the field will stagnate and our systems will remain insecure*

Whither Tech Policy?

Lawyers and Technology

- Lawyers started worrying about technology long before technologists started worrying about the law
  - Privacy work of the NYC Bar Association's Committee on Science and Law started in 1962; the committee itself started in 1959
- Computer scientists: "If we build it, they will come"
  - Who will come? Lawyers? Judges? *Prosecutors?*
  - (As I noted, we had a close call when developing Usenet, and didn't even know it at the time.)

The Link is Now Obvious

- Privacy
- Intellectual property
  - (Major interaction with economics, too)
- Freedom of speech
  - And what of deep fakes? Revenge porn?
- Cross-border issues
- Computer crime
- Liability
- Evidence
- Voting
- AI systems
- The laws of war
- Far, far more

What Do We Need?

- Do we need lawyers with CS degrees?
- Computer scientists with law degrees?
- Both?

It’s hard…

Dynamic Fields

- The law is constantly changing
  - New statutes, regulations, court decisions, etc.
  - In most states, lawyers must take Continuing Legal Education courses every year
- High tech, of course, is even more dynamic—but there's no requirement for continuing CS education
- Conclusion: to work in both fields, you have to make a conscious effort to keep up with change in *both* fields
  - (During the last 10 years, when I should have been learning ML, I was learning law—which is one reason why I do so little modern security work)

CS@CU

One Way to Proceed

- What is your core technical expertise? For me, other than security per se, it was networking (including network security) and software engineering
- What are the legal issues involving that field?
  - Networking: *It's too complicated: How the Internet upends Katz, Smith, and electronic surveillance law.*
  - Software Engineering: *Seeking the source: Criminal defendants' constitutional right to source code*.
- Next: find a lawyer or law prof in that field
- Learn their language, learn the relevant laws, and (especially) court cases

Or…

- Is there a legal problem where CS expertise can help?
- Can CS expertise lend insight?
- Example: a lot of election- and AI-related stuff
- NOTE WELL: “you can’t solve people problems with software”

Reading the Law

- Statutes can be hard to read—they're written in high legalese
  - Just as in programming, precision counts—what do the statutes *actually* say? What are the definitions?
  - Example: in wiretap law, "electronic communication" is data and does not include voice; that's "wire communication", and "oral" is in-person voice… (18 U.S.C. §2510)
- Supreme Court opinions, especially modern ones, are pretty self-contained and rather accessible, but care (and precision) are still needed when reading them
- And of course, understanding the structure of the court system and how cases proceed can also be important

The Legal Issues Can Be Complex

- Reporter: “I’d love to ask you about the impact of current copyright laws on security tasks like the independent red teaming of AI models.”
- Me: “I decline to talk about copyright and AI. To me, the answer seemed obvious, but watching an online discussion between some real lawyers convinced me that that the issues are in fact very subtle.”

You May Not Understand the Law

- There are many aspects that are non-obvious
  - Example: it's very hard to sue over (apparently) illegal NSA surveillance or other privacy violations, because of a legal principle known as "standing"
- Computer scientists love digital signatures, but the law (usually) recognizes "electronic signatures", which are not the same thing
  - 15 U.S.C. §7006(5): "The term "electronic signature" means an electronic sound, symbol, or process, attached to or logically associated with a contract or other record and executed or adopted by a person with the intent to sign the record."
- *Many* more examples

Law Review Articles

- Very different than CS
  - Multiple parallel submissions are *expected*
  - Much longer papers, sometimes upwards of 100 pages
  - Reviews are generally by students, not by peers
  - Submit your CV with a paper, rather than it being anonymous
  - Word, not LaTeX
  - Every factual statement requires a footnote, *with page number*
    - Example: “the range of integers a 32-bit word can hold” needed a citation
  - Court cases are generally cited by *physical page number*
- Many more differences…

So Why Do I Write Law Stuff?

- Other than it's fun?
- Broadening one's skills is always useful
- It's chance to have a different kind of impact on the world
  - Two of my law papers have been cited by appellate courts, once very influentially
- Code does not make law if judges disagree…

So What's the Problem?

- Too often, technical expertise (or the need for it) is not recognized by policy-makers
  - Example: online voting, which virtually every computer scientist who has studied it thinks is a bad idea
  - Example: the need for a technical amicus in the Foreign Intelligence Surveillance Court
- It can be hard to obtain the necessary expertise, especially on a full-time basis
  - The White House has its Office of Science and Technology Policy, but Congress abolished its Office of Technology Assessment (but the GAO has recently brought back some of its functionality)

“One Does Not Simply Walk Into Washington”

- Policy makers and law schools are starting to recognize this
  - Many top law schools have established tech centers
  - A few places, including Stanford, have joint JD/PhD programs in CS
  - There’s a program on Capitol Hill for technology fellows
  - A privacy bill before Congress would create a Bureau of Technology at the FTC
- But: we need better support from university administrations—and funding agencies—for people who teach and do research in both fields
  - What does it take to get tenure?

What’s Next for Me?

I’m Not Retiring to My Rocking Chair

I’m Not Retiring to My Rocking Chair

- Or even to behind my camera
- I intend to continue writing, mostly in law
- But I do have two CS books planned
- Teaching again? Probably not.
- Advising students? I’m not planning on it, but not ruling it out

Red-tailed hawks, East Campus dorm, March 10, 2023

Did I Accomplish Everything I Wanted To?

Did I Accomplish Everything I Wanted To?

- Of course not—no one ever does

Will I Miss Columbia?

Will I Miss Columbia?

- Of course!
- I'll miss my colleagues
- I'll especially miss the many wonderful students I've advised and taught

Why Am I Retiring Now?

- Many reasons, some of them quite personal
- They all add up to “it’s time”
- I made my decision 2.5 years ago and haven’t regretted it at all
- But yes, I’ll miss the students

Students, I love you all!

I’ve Gone Full Circle Between 1972 and Now

CS@CU

“Go, Tony! I throw the torch to you. Your place is the place I occupied. Lead my people. Fight! Live! Become glorious!”

Philip Wylie and Edwin Balmer, *After Worlds Collide*

Questions?