RSACONFERENCE2014

FEBRUARY 24 – 28 | MOSCONE CENTER | SAN FRANCISCO

Share.
Learn.
Secure.

Capitalizing on
Collective Intelligence

Security Education for the new Generation

SESSION ID: MASH-W02
Wednesday, Feb 26, 9:20 AM @ WEST|3018

Jacob West
Chief Technology Officer
HP Enterprise Security Products

Matt Bishop
Professor
University of California, Davis

A Message from Matt

Sorry I cannot be here!

Jacob and I prepared the slides together, and have known each other for years.

This talk represents our views.

Defining security

State of play today

Opportunities

Existing resources

Conclusion

Security Stuff vs. *Secure* Stuff

Security Stuff

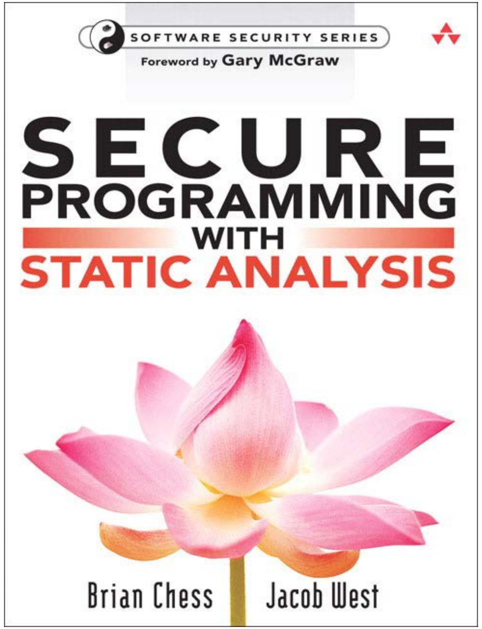
- Responsible for security
- Focus on security activities
- Opportunities for apprenticeship
- Clear career path for the motivated

Doing Stuff Securely

- Responsible for stuff
- Impact on security, but not focus
- Few opportunities to learn
- Hard to map interest to career

What Are We Talking About?

- Robust programming
  - Programming that prevents abnormal termination or unexpected actions
- A “secure” program conforms to a security policy
  - And implicitly requires robustness, but robust programming does not require such conformance
- Here, “secure” is used in the sense of “robust”



Why This Matters

Example:

Exploit a buffer overflow to force a program to do things it should not

- Definitely non-robust
  - Does not handle invalid input properly
- Is it non-secure? That depends if one can…
  - Use the buffer overflow to do things that the security policy disallows (**yes**)
  - Use the buffer overflow to do only things that I could already do (**no**)

Defining security

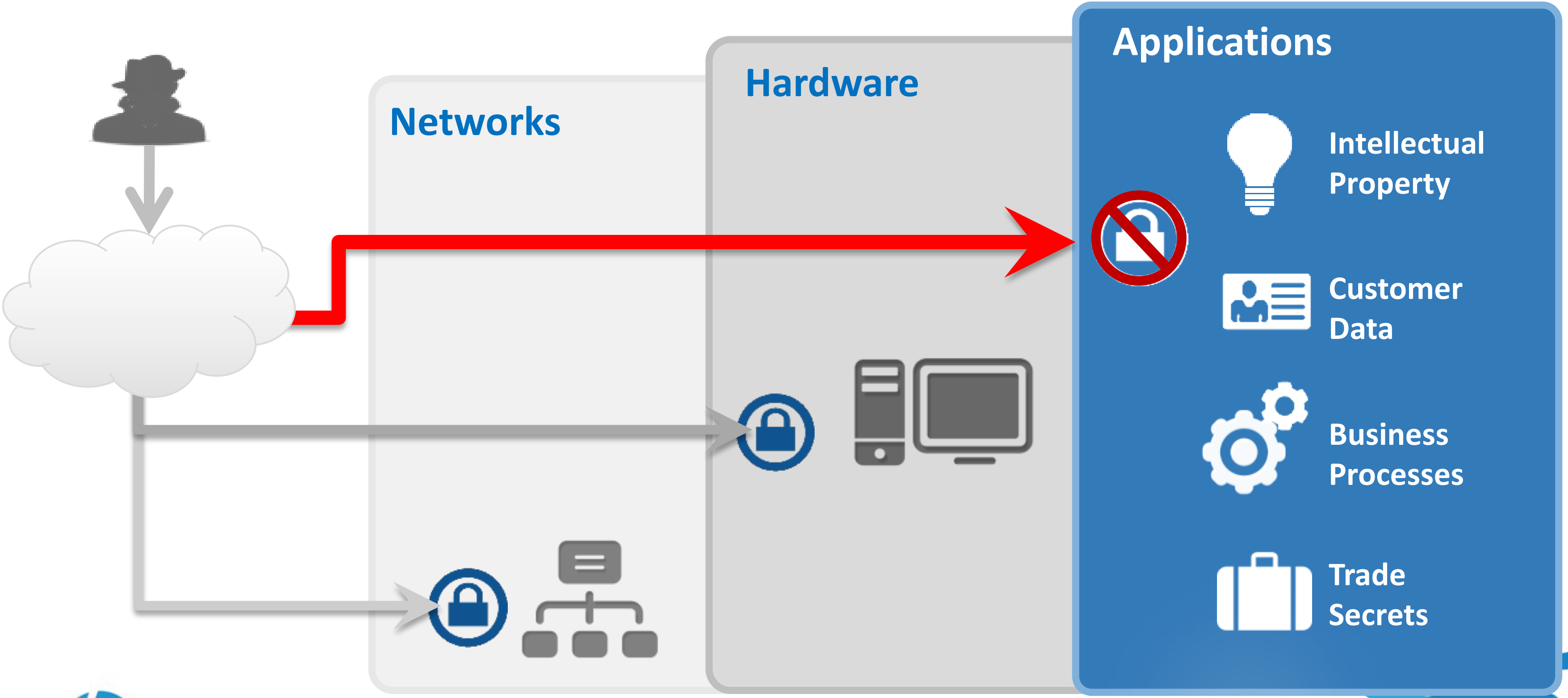
State of play today

Opportunities

Existing resources

Conclusion

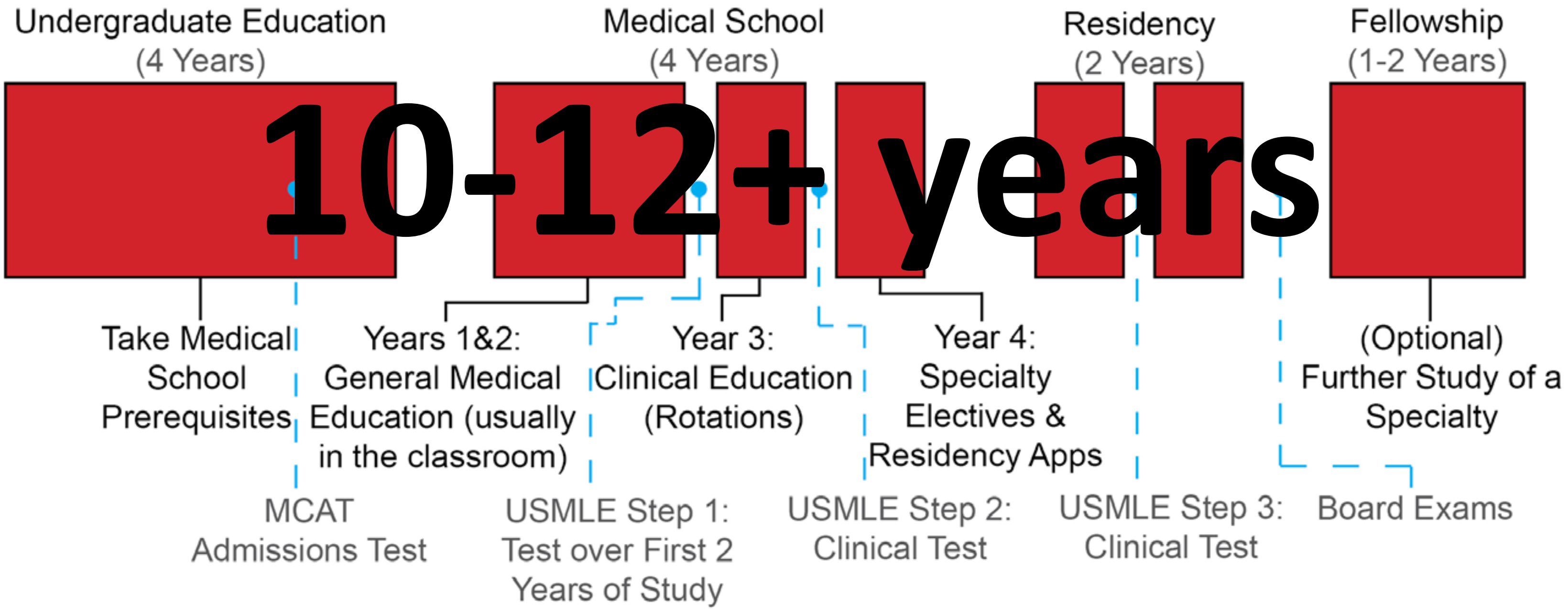
Problem: 84% of Breaches Target Software



Software Security Today

- The line between secure and insecure is often subtle
  - Many seemingly non-security decisions impact security
- Small problems hurt a lot
  - A single bad line of code can put a company in the news
- Smart people make dumb mistakes
  - As a group, programmers repeat the same security mistakes over and over
- We need non-experts to get security right
  - Security and development are both full time jobs

Becoming a Doctor



Undergraduate Education
(4 Years)
Medical School
(4 Years)
Residency
(2 Years)
Fellowship
(1-2 Years)
10-12+ years
Take Medical School Prerequisites
Years 1&2: General Medical Education (usually in the classroom)
Year 3: Clinical Education (Rotations)
Year 4: Specialty Electives & Residency Apps
(Optional) Further Study of a Specialty
MCAT Admissions Test
USMLE Step 1: Test over First 2 Years of Study
USMLE Step 2: Clinical Test
USMLE Step 3: Clinical Test
Board Exams

Becoming a Programmer

Undergraduate Education
(4 Years)

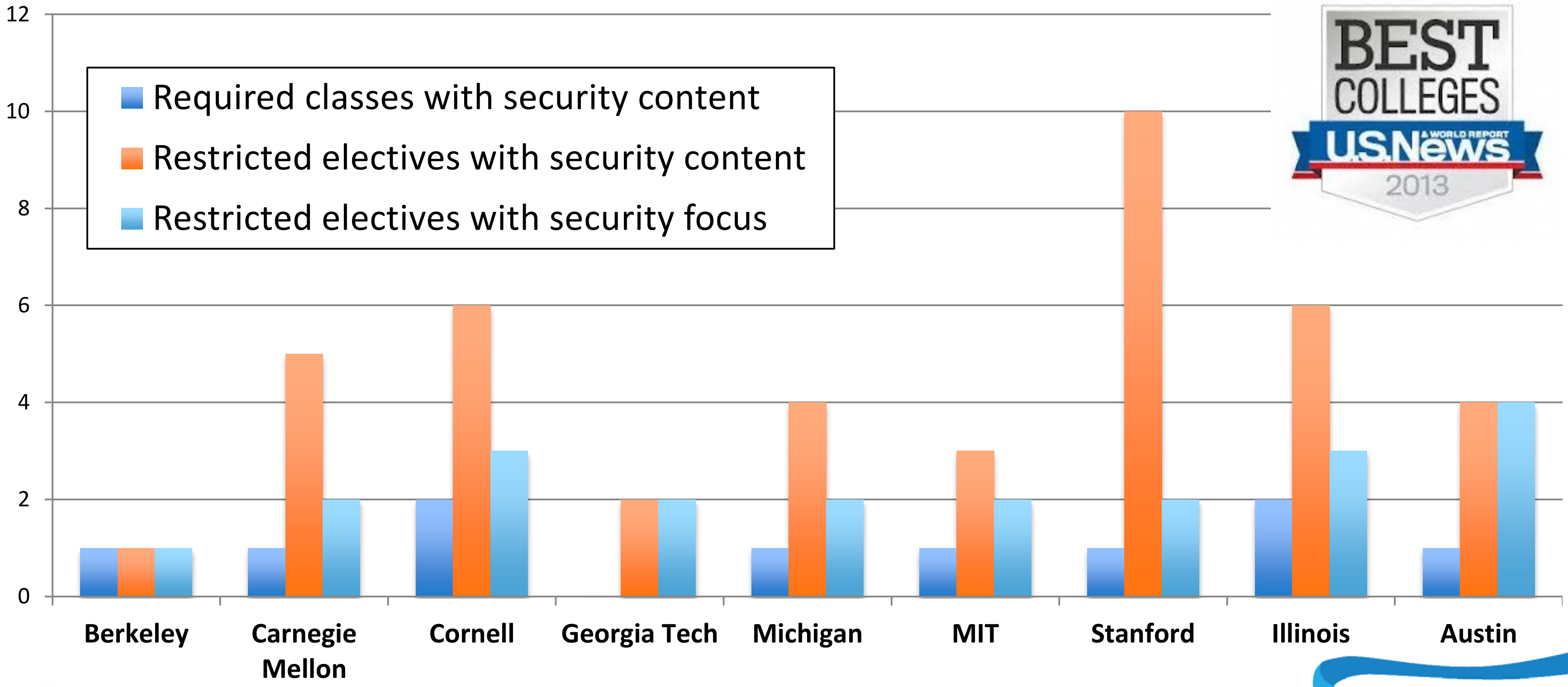
|
Learn
Everything

4 years

Enter Workforce

|
On-the-Job
Training?
Certification?

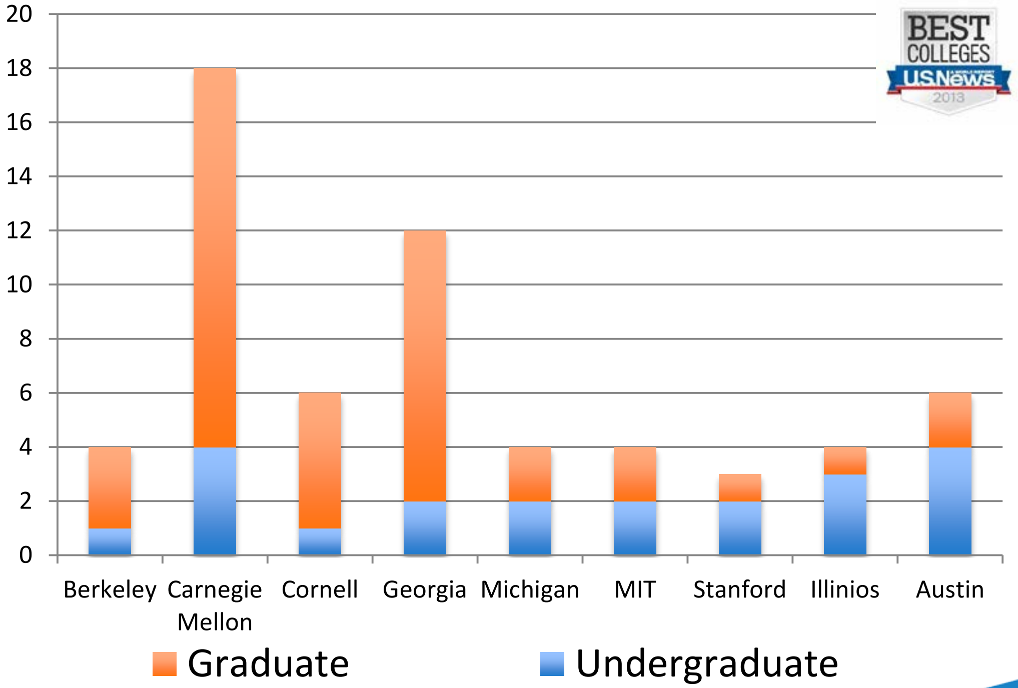
Top 9 Undergraduate Computer Science Programs



BEST COLLEGES U.S.News & WORLD REPORT 2013

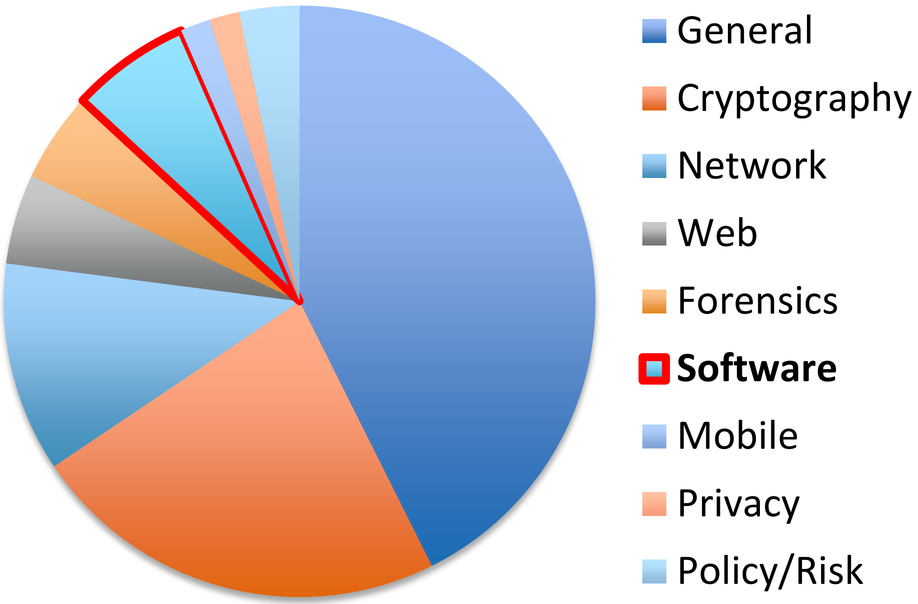
Top 9: Courses with Security as a Focus

- Junior/Seniors specialize
- Only 3 of 9 offer security track
  - Cornell: Security & Trustworthy Systems Track (4 classes)
  - Michigan: Security Track (4 classes)
  - Austin: Information Security Certificate Program (5 classes)



Top 9: Courses Focus on Traditional Security

Software security appears at 3 of 9



The Fundamental Problem

- We don’t write software that is robust
  - Some exceptions in special cases
- We don’t build systems to meet security requirements
- Many different models for developing software

What Will Drive Improvement?

Commercial

- Financial savings
  (avoid cleaning up messes)
- Simpler maintenance
- Improved reputation

- Software liability?

Government

- Financial savings
  (avoid cleaning up messes)
- Simpler maintenance
- National security

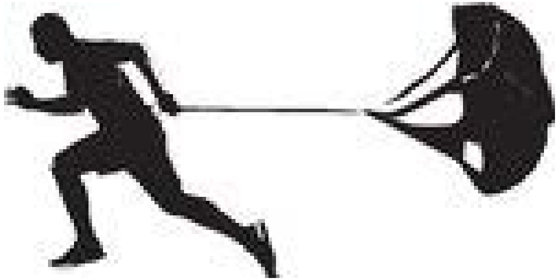
Software Liability

- You can't say "I'm not responsible for anything"
  - Chain of distribution (e.g. supply chain) liability exists now
- You can limit liability somewhat by defining use and environment
  - Then you're liable in that context but (probably) not in others
- It *is* coming . . .
  - EULAs may not be enforceable (*adhesion contracts*)

So What’s Holding Us Back?

Commercial and Government

- Need to spend more money
- Longer time-to-market
- No legal liability for bad software
- Need to pay more attention to installation, maintenance, and use
- Lack of people to write good code



So What's Holding Us Back?

Academia

- Robust coding not seen as integral to programming
  - Textbooks *loaded* with examples of non-robust programming
- Lack of support for *enforcing* and *grading* for robust coding
  - Ties into lack of graders who really know about this
- Lack of faculty who understand robust programming
  - And intimidation factor for those who *know* they don't understand it

Lack of Resources

Assurance costs!

- Industry expected to deliver secure, robust products without resources for the extra effort required to deliver them
- Academia expected to teach *and reinforce* robust programming without resources for the extra effort in supporting this

Lack of People

- Need to teach people how to write robust programming
  - Need to emphasize the *practice*, both in education *and* industry
- Continuous practice is *key* to reinforcing, maintaining, extending skills

Defining security

State of play today

Opportunities

Existing resources

Conclusion

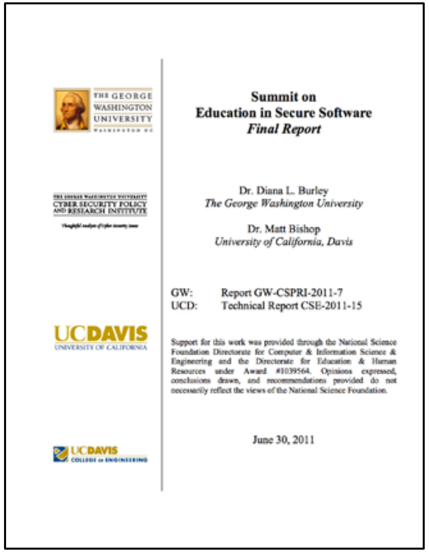
Focus for Rest of Talk

Everyone lacks resources!

- How can industry and government work with academia?
  - Carrots, not sticks
  - Security tuned to environment and use
  - What is “secure” varies among companies and government organizations

What Do We Need to Teach Secure Programming?

- *Summit on Education in Secure Software* (SESS)
  - Diana Burley
    The George Washington University
  - Matt Bishop
    University of California at Davis
- Funded by NSF
- 60 participants: academia / industry / government
- nob.cs.ucdavis.edu/~bishop/notes/2011-sess/2011-sess.pdf



Summit on
Education in Secure Software
Final Report

Dr. Diana L. Burley
The George Washington University

Dr. Matt Bishop
University of California, Davis

GW: Report GW-CSPRI-2011-7
UCD: Technical Report CSE-2011-15

Support for this work was provided through the National Science Foundation Directorate for Computer & Information Science & Engineering and the Directorate for Education & Human Resources under Award #1039564. Opinions expressed, conclusions drawn, and recommendations provided do not necessarily reflect the views of the National Science Foundation.

June 30, 2011

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON DC
THE GEORGE WASHINGTON UNIVERSITY
CYBER SECURITY POLICY AND RESEARCH INSTITUTE
UCDAVIS
UNIVERSITY OF CALIFORNIA
UCDAVIS
COLLEGE of ENGINEERING

SESS Objectives

- Engage stakeholders from academia, industry, and government to discuss teaching secure programming
- Use discussion as basis for a collaborative effort to develop a comprehensive agenda for secure software education

Recommendations

- Increase faculty who understand the importance of secure coding
  - Establish professional development opportunities for faculty
- Integrate computer security content into existing courses
  - Provide faculty support for the inclusion of security content
- Require at least one computer security course for all college students

Recommendations

- Promote collaborative problem solving and solution sharing
  - Encourage partnerships and collaborative curriculum development
- Use innovative teaching methods to strengthen the foundation of computer security knowledge
  - Develop metrics to assess progress toward meeting educational goals
- Highlight the role that computer security professionals should play in key business decision making processes

Summary

- Holistic view of secure education suggests programmers and non-programmers alike must be educated in the core principles
- Structural enablers
  - Cultural shift among faculty and industry stakeholders that supports the development of a holistic view of software security
  - Identification of measurable objectives and corresponding measurements
  - Development of national licensure programs
  - Alignment of expectations for university education and realistic

What Can Academia Do?

- Include robustness in evaluation of programs and projects
- Create a “secure programming clinic”
  - Like an English clinic, or a writing clinic for law schools
- Provide supplementary material for textbooks, classes
  - These should emphasize robust programming

What Can Industry Do?

- Key is to *show* more than *say* secure development is important
- Make clear that the skills are important for hiring
    - Mention their need in job openings
    - Preference to those with skill in this also helps

Experience
Experience
observing, encour
practical knowledge, skill,
resulted in understanding
of events participated in

Work With Students and Faculty

- Internships
  - Students *love* these; good recruiting tool
  - Tasks requiring robust programming emphasize its importance to students
- Help teach students
  - Review students’ code
  - Team with colleges in senior/capstone projects

What Will This Do?

- Increase student demand
  - If students see it as important, they will ask about it in class, evaluate programs, faculty in part on it
- Increase your visibility
  - Good recruiting tools
  - A corporate “good citizen”

Government Support

- Act like an industry
- Government can also fund programs
  (e.g. DHS and the Software Assurance Curriculum Project)
- Programs should support future faculty (as well as engineers)
  - People willing to commit to teaching
- *Imperative: target funding towards this specific purpose*
  - <u>Require</u> funding to be used for supporting robust programming
  - If done as adjunct, likely to disappear in the main purpose of the funding

Defining <u>security</u>

State of play today

Opportunities

Existing resources

Conclusion

Existing Resources

- Industry academic enablement programs
- Tradeshows and conferences
- Competitions and contests
- Training and certification
- Specialized university programs

Industry Academic Enablement Programs

Focused on facilitating university education in security

Notable Programs

- Cisco
- Hewlett-Packard
- IBM
- Microsoft

Delivery Methods

- Direct to students or young professionals with certification programs
- Define a methodology, enlist industry specialists to assist with delivery
- Collaborate with universities and non-profits on research

Tradeshows and Conferences

Notable Conferences

- IEEE Symposium on Security and Privacy
- ACM Computer and Communications Security
- “The Colloquium”
  (Colloquium on Info. System Security Education)
- USENIX Security Symposium

Goals

- Share advancements in research
- Enhance with curricula security-centric topics

Competitions and Contests

Notable Competitions

- CyberPatriot
- National Collegiate Cyber Defense Competition
- UCSB iCTF
- Cyber Security Awareness Week (CSAW)
- DEFCON

CYBERPATRIOT
NATIONAL COLLEGIATE CYBER DEFENSE COMPETITION
CSAW'13
Cyber Security Awareness Week @ NYU-Poly
UNIVERSITY · OF · CALIFORNIA · SANTA BARBARA
LET THERE BE LIGHT
DEFCON

Why Capture-the-Flag?

- Goal-oriented and rewards both participation and success
- Opportunity to network with peers and industry professionals

Training and Certification

Notable Programs

- (ISC)^2
- SANS Institute
- CompTIA

Motivations and Objectives

- Industry effort to develop and ensure baseline skillsets
- Differentiate candidates for human resources and hiring managers
- Validate and recertify relevant security experience

Scholarship Programs

Government Scholarships (for work commitment)

- CyberCorp: Scholarships for Service
- DOD Information Assurance Scholarship Program

Private Scholarships

- (ISC)^2 Scholarships (Community College, Undergrad, Grad)
- Armed Forces Communications and Electronics Association (Community College, Undergrad, Grad)
- National Security Scholars Program (Undergrad)
- Symantec Graduate Fellowship (Grad)
- Applied Computer Security Associates (Undergrad, Grad)

Scholarship for Women Studying Information Security

Support women with a demonstrated interest in security, through coursework, internships or work experience to complete a Bachelors or Masters degree

- Must be entering junior or senior year of Bachelors or first year of Masters
- Administered by *Applied Computer Security Associates* (ACSA) and *Committee on the Status of Women in Computer Research* (CRA-W)
  - Awarded a single $10k scholarship annually pre-2014
- Includes attendance at ACSA, CRAW, as well as internship opportunities

RSACONFERENCE2014

FEBRUARY 24 – 28 | MOSCONE CENTER | SAN FRANCISCO

Defining security

State of play today

Opportunities

Existing resources

Conclusion

Many Myths

- Myths about universities confuse how academia can put teaching secure programming into practice
  - *There is no room…*
  - *If students learn to write secure programs…*
  - *Academic institutions are…*
  - *We know what to do…*
- And some questionable ideas don’t help either
  - Testing students knowledge
  - Unsupported mandates

Myth #1

There is no room in curricula for a course on secure programming

- You don’t need a separate course
- Simply check programs submitted during all courses for robustness
  - Make writing robust programs beneficial (through grades or other mechanisms)
  - Provide resources so students can see how to do this, or get help to do it

Myth #2

If students learn to write secure programs, the state of software and system security will dramatically improve

- Will companies accept increased cost, time to market?
- Will customers pay higher prices, endure longer development times?
- Will students be encouraged (required) to practice what they learned?

Myth #3

Academic institutions are hierarchical in organization

- Implication is that deans, provosts, presidents can order this taught
- Learning styles, environments differ
- May not be a 'best' or 'right' way to teach this

Myth #4

We know what to do and how to do it

- We have ideas, but don't know
  - Needed: research on education
  - Needed: funding, people to do this
- The Summit on *Education in Secure Software* suggested ways to do this
  - SESS results are general
  - Approaches must be tailored to various environments

Questionable Idea #1

Testing students' knowledge

- Who creates the tests?
- Who is being tested?
- How do you know that you are testing what is important? (that is, the "right thing")
- Who determines what is an acceptable result?
- Teaching to the test, rather than to learn the material

Questionable Idea #2

Unsupported mandates

- The support has to come from somewhere
  - It’s like a zero-sum game
- What do you want to weaken?
  - If you only have so many resources, something will have to give
  - You don’t want to weaken the core foundation of understanding *why* certain programming paradigms are critical

Conclusion

“We must all hang together, or
we shall all hang separately.”
- Benjamin Franklin

- The state of practice can, and must, change
- Understand that academia is a different environment—completely
- Teaching robust programming, *and nothing more*, will not help
- The marketplace must also change, as must current practice
- The public will be the main driver (unfortunately, probably with lawsuits)

Questions?

Jacob West
jbw@hp.com
Chief Technology Officer
HP Enterprise Security Products

Matt Bishop
mabishop@ucdavis.edu
Professor
University of California, Davis