Using the CS2013 Body of Knowledge to Reflect on our Security Courses

Abstract. The ACM/IEEE Computing Curriculum 2013 is a community effort with representation from Academia and industry to outline curricular recommendations for undergraduate Computer Science degree programs. The effort began in 1968 [1] and conducts a complete review every ten years. The previous complete review was completed in 2001. [2] The current 2013 review is being developed to incorporate rapidly changing topics as technology and the world's use of technology evolves; however must do so within the curricular constraints of a complete undergraduate curriculum. This effort describes the architecture of CS2013 and the lessons learned in mapping our curriculum against the CS2013 computing curriculum.

1 Introduction

The world that our undergraduates must be prepared to succeed in is changing rapidly. Topics or study areas that seemed critical when a student began his or her undergraduate program may be out of date or encompasses by another emerging topic by the time they graduate. The Association for Computing Machinery and the Institute of Electrical and Electronics Engineers Computer Society (IEEE-CS) has a long standing interest in providing input into the educational programs, teaching the future professionals in the field. Since 1968 [1], a joint commission from the two bodies has created a set of curricular recommendations for institutions to use in shaping the Computer Science undergraduate curriculum. This effort has traditionally been fully reviewed every 10 years with a minor interim assessment at the five year mark. The last major review was completed in 2001 and the interim review was completed in 2008. [2][3] Each new version of the ACM/ IEEE Computing Curriculum provides an opportunity for undergraduate Computer Science programs to review and assess their curriculum against a community constructed set of objectives. Our institution began a review using the CS2013 strawman, a draft version of the new set of the curriculum recommendations. In this paper, we begin in section two by providing further discussion of the CS2013 process and structure with a focus on the Information Assurance and Security knowledge area. In section three, we describe the effort at our institution to review the CS2013 strawman and our process to conduct a curriculum review. In section four we conclude with recommendations.

2 A Review of the CS2013

The ACM/ IEEE Computer Science 2013 followed the development model of previous years. Two co-chairs were appointed by the ACM and the IEEE who then created a steering committee; of a combined body of professionals from academia and industry, to review the current state of undergraduate computer science education and the new and emerging requirements. The CS2013 effort began with a review of the 2001 and 2008 IEEE/ACM curriculum reviews and the generation of a survey to be completed by current Computer Science Department chairs. The survey was distributed to approximately 3500 Computer Science (and related 105 discipline) Department Chairs and Directors of Undergraduate Studies (1500 in the United States and 2000 internationally); 201 responses were received. A detailed analysis of the report is available in the CS2013 ironman report. [4] This input, along with the 2001/2008 curriculum reviews, was critical in establishing the goals for the 2013 document.

2.1 CS2013 goals

The steering committees used the feedback from the survey and input from critiques of the CS2001/2008 recommendations to review the body of knowledge requirements for an undergraduate computer science graduate. The analysis resulted in the establishment of 10 goals:

1. Computer Science curricula should be designed to provide students with the flexibility to work across many disciplines.
2. Computer Science curricula should be designed to prepare graduates for a variety of professions, attracting the full range of talent to the field
3. CS2013 should provide guidance for the expected level of mastery of topics by graduates.
4. CS 2013 must provide realistic, adoptable recommendations that provide guidance and flexibility, allowing curricular designs that are innovative and track recent developments in the field.
5. The CS2013 guidelines must be relevant to a variety of institutions.
6. The size of the essential knowledge must be managed.
7. Computer Science curricula should be designed to prepare graduates to succeed in a rapidly changing field.
8. CS2013 should identify the fundamental skills and knowledge that all computer science graduates should possess while providing the greatest flexibility in selecting topics.
9. CS2013 should provide the greatest flexibility in organizing topics into courses and curricula.
10. The development and review of CS2013 must be broadly based.

2.2 CS2013 architecture

The knowledge of body was reviewed and separated out into knowledge areas based on the review and feedback process. The steering committee was organized into subcommittees, one for each knowledge area. The knowledge areas are topical collections of similar material. Knowledge areas are explicitly not courses. For example Operating Systems is a knowledge area, however the coverage is about topics that are relevant to Operating System concepts which might be presented in a course under a different title. The 18 knowledge areas are:

- AL-Algorithms and Complexity
- AR-Architecture and Organization
- CN-Computational Science
- DS-Discrete Structures
- GV-Graphics and Visual Computing
- HCI-Human-Computer Interaction
- IAS-Security and Information Assurance (new in 2013)
- IM-Information Management
- IS-Intelligent Systems
- NC-Networking and Communication (new in 2013)
- OS-Operating Systems
- PBD-Platform-based Development (new in 2013)
- PD-Parallel and Distributed Computing (new in 2013)
- PL-Programming Languages
- SDF-Software Development Fundamentals (new in 2013)
- SE-Software Engineering
- SF-Systems Fundamentals (new in 2013)
- SP-Social Issues and Professional Practice

Each knowledge area is further broken down into knowledge units. The review of each knowledge area is beyond scope for this paper; for reference please see the CS2013 ironman draft. [4]

The knowledge areas are further broken down into knowledge units (KU) that are comprised of naturally grouped topic areas. For example, within the Information Assurance and Security knowledge area, example knowledge units include network security and cryptography. Each of the KU's contains a list of topics that are relevant to that KU (further detail is provided in paragraph 2.3). The Information Assurance and Security (IAS) knowledge area is unique among the knowledge areas due to its almost cross cutting relevancy to the other knowledge areas (KA).

To detail the depth of coverage of a given KA/KU, the CS2013 architecture further defines the concept of core-tier 1, core-tier 2, and elective KUs. Core-tier 1 and core-tier 2 requirements are represented in a quantity of lecture hours. For an example using the IAS KA, please see Table 1. Core-tier 1 hours are expected to be present in any undergraduate curriculum. Approximately 80% of core-tier 2 hours are expected to present. The flexibility in core-tier 2 hours is intended to reflect the variability in specialties across academic institutions.

2.3 CS2013 Information Assurance and Security knowledge area

Information assurance has been defined as "a set of controls (technical and policy) intended to protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. This includes providing for restoration of information systems by incorporating protection, detection, and reaction capabilities."[5] [6]. This concept of assurance also carries a connotation of an attestation that past processes or data is valid.

Information assurance and security education, then, includes all efforts to prepare a workforce with the needed knowledge, skills, and abilities to assure our information systems and attest to the validity of the current state of processes and data. Information assurance and security education has been growing in importance and activity for the past two decades.

The McCumber model [7] [8] has been widely used over the past 10 years to broadly define the relationships between information states, security services, and security operations (called counter measures in the original model). As the security and assurance landscape has matured, the relationships should be clarified to address that the full spectrum operational aspect of security is not correctly contained by the term counter measures and that emergence of assurance of processes (current and past) is playing a critical role in the field of IAS.

The aim of this KA is to define the core (tier1 and tier2) and elective knowledge threads that depict what a computer science undergraduate should possess upon matriculation.

The IAS KA is unique in the collection of KA's due to its pervasive nature in all KA's. One can express this cross cutting impact by comparing security to performance. In the past, many concepts in Computer Science were performance based. Algorithms were developed to increase the performance of memory utilization or database searches. In this light, the way we do things in Computer Science must be done securely. This is not to say that Information Assurance and Security does not have concepts that belong solely to the IAS KA. As shown in table 1, concepts that are unique to IAS listed with specific core tier 1 and tier 2 hours.

Table 1: IAS hours Distribution

Knowledge Unit	Core-Tier1 hours	Core-Tier2 hours	Electives
IAS/Foundational Concepts in Security	1	-	N
IAS/Principles of Secure Design	1	1	N
IAS/Defensive Programming	1	1	Y
IAS/Threats and Attacks	-	1	N
IAS/Network Security	-	2	Y
IAS/Cryptography	-	1	N
IAS/Web Security	-	-	Y
IAS/Platform Security	-	-	Y
IAS/Security Policy and Governance	-	-	Y
IAS / Digital Forensics	-	-	Y
IAS/Secure Software Engineering	-	-	Y

Each knowledge unit contains a collection of topics. For the knowledge unit, "IAS/Principles of Secure Design", the topics are shown below. Each topic has an associated learning outcome with a desired level of comprehension (Familiarity, Application, and Usage). A detailed listing of the KU's and Topics are available in the CS2013 IronMan. [4]

IAS/Principles of Secure Design
[1 Core-Tier1 hours, 1 Core-Tier2 hours]
Topics:

[Core-Tier1]

- Least privilege and isolation (cross-reference OS/Security and Protection/Policy/mechanism separation and SF/Virtualization and Isolation/Rationale for protection and predictable performance and PL/Language Translation and Execution/Memory management)
- Fail-safe defaults (cross-reference SE/Software Construction/ Coding practices: techniques, idioms/patterns, mechanisms for building quality programs and SDF/Development Methods/Programming correctness)
- Open design (cross-reference SE/Software Evolution/ Software development in the context of large, pre-existing code bases)
- End-to-end security (cross reference SF/Reliability through Redundancy/ How errors increase the longer the distance between the communicating entities; the end-to-end principle)
- Defense in depth
- Security by design (cross reference SE/Software Design/System design principles)
- Tensions between security and other design goals

[Core-Tier 2]

- Complete mediation
- Use of vetted security components
- Economy of mechanism (reducing trusted computing base, minimize attack surface) (cross reference SE/Software Design/System design principles and SE/Software Construction/Development context: "green field" vs. existing code base)
- Usable security (cross reference HCI/Foundations/Cognitive models that inform interaction design)
- Security composability
- Prevention, detection, and deterrence (cross reference SF/Reliability through Redundancy/Distinction between bugs and faults and NC/Reliable Data Delivery/Error control and NC/Reliable Data Delivery/Flow control)

Learning outcomes:

[Core-Tier1]

1. Describe the principle of least privilege and isolation and apply to system design [application]
2. Understand the principle of fail-safe and deny-by-default [familiarity]
3. Understand not to rely on the secrecy of design for security (but also that open design alone does not imply security) [familiarity]
4. Understand the goals of end-to-end data security [familiarity]
5. Understand the benefits of having multiple layers of defenses [familiarity]
6. Understand that security has to be a consideration from the point of initial design and throughout the lifecycle of a product [familiarity]
7. Understanding that security imposes costs and tradeoffs [familiarity]

[Core-Tier2]

8. Describe the concept of mediation and the principle of complete mediation [application]
9. Know to use standard components for security operations, instead of re-inventing fundamentals operations [familiarity]
10. Understand the concept of trusted computing including trusted computing base and attack surface and the principle of minimizing trusted computing base [application]
11. Understand the importance of usability in security mechanism design [familiarity]
12. Understand that security does not compose by default; security issues can arise at boundaries between multiple components [familiarity]
13. Understand the different roles of prevention mechanisms and detection/deterrence mechanisms [familiarity]

As discussed previously, the topics in IAS run as a common thread through all of the CS2013 KA's. While there are KU and topics that belong uniquely to IAS, the significant majority of the topics relevant to security and assurance are actually distributed widely in the emailing KA's. Table 2 shows the distribution of hours that are "security relevant". As can be seen, while IAS has 9 combined core hours, there are 63.5 hours distributed through the other KA's.

Table 2: IAS Cross KA Hour Distribution

KA's	Core-Tier1 hours	Core-Tier2 hours	Elective
IAS	3	6	Y
IAS distributed in other KA's	32	31.5	Y

Two examples of nature in which Information Security and Assurance topics are found in other KA's within the CS2013 body of knowledge the Operating System (OS) and System Development Fundamentals (SDF). The topic objectives are omitted however can be found in [4].

OS/Security and Protection [2 Core-Tier2 hours]
Topics:

- Overview of system security
- Policy/mechanism separation
- Security methods and devices
- Protection, access control, and authentication
- Backups

SDF/Development Methods [10 Core-Tier1 hours]
Topics:

- Program comprehension
- Program correctness
    - Types or errors (syntax, logic, run-time)
    - The concept of a specification
    - Defensive programming (e.g. secure coding, exception handling)
    - Code reviews
    - Testing fundamentals and test-case generation
    - Test-driven development
    - The role and the use of contracts, including pre- and post-conditions
    - Unit testing
- Simple refactoring
- Modern programming environments
    - Code search
    - Programming using library components and their APIs
- Debugging strategies
- Documentation and program style

3 A Curricular Mapping

Two of the authors, who were not participants in the curricular developmental effort for the Strawman, wanted to see how the current state of our Computer Science Program compared. We conducted a self assessment to reflect on how well the program followed these recommendations. We were not interested in achieving convergence with the forthcoming Body of Knowledge, but rather wanted to frame an introspective look at our program using the metrics developed by the collective wisdom of the panel. After an initial look at the Body of Knowledge, we suspected that we would not be close to meeting the recommended hours per topic, and were concerned that our courses would not line up with the knowledge areas.

3.1 Create a tool.

As the first step, we started with the courses that we teach and counted the lecture hours spent on topics listed in the Body of Knowledge. We saw that, while some courses contributed almost exclusively to a single knowledge area, others' contributions spread across several. Accurately mapping the full Computer Science program across all knowledge areas by hand would be a difficult undertaking. We needed a tool to effectively conduct our assessment and thus copied the curricular recommendations into a spreadsheet. We made a separate worksheet for each knowledge area, with the groupings and topics in rows and each of our courses in the columns of the worksheet. This allowed us to enter the number of lecture hours devoted to each topic for each course. The spreadsheet

tallied our results for summary views, such as coverage of Tier-1 topics and knowledge area coverage by course.

3.2 Course Director Interviews.

We sat down with each course director teaching a required Computer Science course and assisted them in conducting the knowledge area mapping for their course. We did not include the elective courses in our assessment because we wanted to capture the baseline common experience of our majors. We guided each course director to the worksheet thought to contain a high density of topics for their course, and also highlighted other related worksheets. Some of the course directors voiced concern that their classroom facilitation was under-represented based on the lecture hours metric as defined by the CS2013. All classes at our institution are capped at eighteen students per class, and our pedagogy often involves instructor supervised activities, such as board-work and other structured events, including a week-long Cyber Defense Exercise. We were puzzled at how to assess the hours for the networking requirement for our program, since students must elect to take one of three offered networking courses. We assessed each of these three courses independently, and then took the intersection of the results to derive the number of hours we could be confident that all of our students received. Guided by the language in the learning outcomes and topics, each course director quickly recognized specific concepts taught in their courses and they were able to complete their interview within 20-30 minutes per course.

We observed that each of our 40-lesson courses contributed approximately 25-30 hours to the CS2013 requirements, and re-engaged course directors when anomalous results were observed, resulting in a few corrections and clarifications. We noted that we significantly exceeded the recommended coverage for some topics, most notably in the Information Management Concepts (Databases) and Algorithms and Complexity (Automata Theory) knowledge areas. In other areas, our topic hours fell short of the recommendations, sometimes surprisingly.

3.3 Faculty Review Meeting.

To gain a better interpretation of our results we brought our faculty together for a review of the findings. This meeting was eventful and beneficial to our program. Everyone was surprised that we fell short on the recommendation for the Information Assurance and Security and Networking and Communication knowledge areas, especially since we regard these areas as strengths in our program. The group recognized a need to increase coverage of parallelism and concurrency in the Program, and we are currently in the process of implementing curricular changes. The process also helped us to identify a course that contributed very few hours to meeting the CS2013 requirements – in an already busy program this became an obvious course to cut from the requirements for future students.

We determined that our decision to take the intersection of the required Networking electives does not reflect the actual experience of our students. Nearly all of our majors elect to take a cyber defense course that covers the knowledge area in addition to a second networking elective that adds depth in the knowledge area. Furthermore, we neglected to count an introductory IT course that all students take in their first year. There was a strong concurrence that we should continue our level of coverage for many of the instances where topics were covered in greater than recommended levels. There was a general consensus that usefulness of the assessment was diminished based on solely using the metric of lecture hours to tally hours of coverage per topic. However, we believe that periodic reassessments will be easy to conduct and will provide valuable insight and direction for our program.

3.4 Results.

The review of the CS2013 body of knowledge against the current curriculum was a beneficial effort. Through the review, we identified areas in the curriculum where our current focus exceeds the body of knowledge and areas where additional assessment is needed. The courses where we found the majority of our IAS related curricular cover are:

- IT105 Introduction to Information Technology
- CS400 Computer Science Seminar
- CS401 Software Systems Design
- CS403 Object Oriented Concepts
- CS478 Analysis of Programming Languages
- CS482 Cyber Security

Table 3 depicts the hour coverage of the IAS knowledge units within our undergraduate computer science curriculum. The hours are denoted as (core-tier 1 | core-tier 2 | elective). The total coverage amounted to 12 core-tier 1 hours, 17.5 core-tier 2 hours, and 8 elective hours.

Table 3: Course Coverage of IAS KA

Knowledge Unit	**IT105**	**CS400/ CS401**	**CS403/ CS478**	**CS482**
IAS/Foundational Concepts in Security	1|0|0	2|0|0	-	2|0|0
IAS/Principles of Secure Design	-	1|1|0	1|0|0	2|0|0
IAS/Defensive Programming	-	-	2|0|0	1|0|0
IAS/Threats and Attacks	0|2|0	0|1|0	-	0|4|0
IAS/Network Security	0|1|0	-	-	0|6|0
IAS/Cryptography	0|.5|0	-	-	0|2|0
IAS/Web Security	-	-	-	0|0|2
IAS/Platform Security	-	-	-	0|0|5
IAS/Security Policy and Governance	-	-	-	-
IAS / Digital Forensics	-	-	-	0|0|1
IAS/Secure Software Engineering	-	-	-	-

4 Conclusions and Recommendations

We found that conducting the review of our program with the CS2013 guidance was a time consuming task, but worth the effort. We can offer a few concise recommendations for anyone considering a similar endeavor:

- Our divide-and-conquer approach enabled us to efficiently collect the data while the faculty review meeting was vital to understanding the results.
- The tool we created allows for rapid visualization of the results through tallying and charts, and for easy re-use of the study in future years.
- Use the provided metric of lecture hours in CS2013 as a starting point, but establish a meaningful definition of hours for your unique institution.
- A reasonable method is needed to capture the contributions from electives to fully capture the experience of the majority of students.
- Discourage the view that difference is bad. Well-explained differences make programs unique and discussing the reasons for those differences may serve to strengthen a program's vision.

In addition to the benefit of any bottom up curricular review, the assessment with special attention paid to where security is integrated into the breath of courses will result in the introduction of security topics much earlier in the courses offered in our computer science program.

5 References

1. ACM Curriculum Committee on Computer Science. 1968. Curriculum 68: 213 Recommendations for Academic Programs in Computer Science. *Comm. ACM* 11, 3 214 (Mar. 1968), 151-197.

2. ACM/IEEE-CS Joint Task Force on Computing Curricula. 2001. ACM/IEEE Computing Curricula 2001 Final Report. http://www.acm.org/sigcse/cc2001. 217

3. ACM/IEEE-CS Joint Interim Review Task Force. 2008. Computer Science Curriculum 221 2008: An Interim Revision of CS 2001, Report from the Interim Review Task Force. 222 http://www.acm.org/education/curricula/ComputerScience2008.pdf

4. ACM/IEEE-CS CS2013 IronMan draft, http://ai.stanford.edu/users/sahami/CS2013/ironman-draft/cs2013-ironman-v1.0.pdf

5. National Security Agency, http://www.nsa.gov/ia/iaFAQ.cfm?MenuID=10#1

6. NIST publication 800-53. http://csrc.nist.gov/publications/nistpubs/800-53-Rev2/sp800-53-rev2-final.pdf

7. Machonachy, W. Victor; Schou, Corey D.; Ragsdale, Daniel; Welch , Don; "A model for Information Assurance: An Integrated Approach", Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, United States Military Academy, West Point , NY, 5-6 June 2001

8. McCumber, John. "Information Systems Security: A Comprehensive Model". Proceedings 14th National Computer Security Conference. National Institute of Standards and Technology. Baltimore, MD. October 1991