Development of Peer Instruction Questions for Cybersecurity Education

William E. Johnson, Allison Luzader, Irfan Ahmed, Vassil Roussev, Golden G. Richard III
University of New Orleans

Cynthia B. Lee
Stanford University

Abstract

Cybersecurity classes should be focused on building practical skills along with the development of the open mindset that is essential to tackle the dynamic cybersecurity landscape. Unfortunately, traditional lecture-style teaching is a poor match for this task. Peer instruction is a non-traditional, active learning approach that has proven to be effective in many fundamental courses of computer science. The main challenge for faculty in adopting peer instruction is the development of conceptual questions. This paper presents a methodology for developing peer instruction questions systematically for cybersecurity courses. The method consists of four stages: concept identification, concept trigger, question presentation, and question development. The paper further provides an analysis of 172 questions developed over the period of ten months by the authors for two cybersecurity courses: introduction to computer security and network penetration testing. Finally, it discusses four examples of peer instruction questions in the context of the aforementioned methodology.

1 Introduction

Cybersecurity is one of most strategically important areas of computer science and also a difficult discipline to teach effectively. The escalating reliance on IT tools in all aspects of social life is leading to ever increasing costs in cybersecurity failures. The majority of these failures are the result of poor understanding of the security landscape, an overly abstract view of important computing concepts, and an inability to adapt to new threats.

Engineering a secure IT system, in addition to technical skills, requires out-of-the-box thinking that takes into account the incentives and capabilities of both the attacker and the defender. To be effective, a cybersecurity professional must be flexible and creative, able to quickly adapt within the fast-changing security landscape. In such a dynamic environment, education is a continuous process and requires the mindset that learning on the job is part of the daily routine. It is imperative that we find methodologies that can reliably improve learning outcomes and develop workforce proficiency in these strategically important areas.

Unfortunately, the traditional lecture is a poor match for the need to develop students into creative thinkers and lifelong learners, and this is especially true for cybersecurity education—both within and outside of academia. This is the direct result of an over-emphasis on *specific* (lifespan limited) technical skills without attention to fundamental conceptual underpinnings. Other challenges include a lack of technical depth, and impatience towards developing broader analytical skills.

One of the main difficulties in delivering the necessary educational outcomes is that students need to experience a significant number of realistic situations before they can appreciate practical security problems and start to reason about the corresponding situations. In other words, presenting the underlying concepts is a necessary part of the job for the instructor, but it is well short of sufficient. One of the biggest instructional challenges is to balance the requirements of discussing concepts and building hands-on skills within the confines of a semester. In our view, the only way to accomplish this is to encourage the students to do more preparation *before* coming to class (and to continue this preparation well after class has ended), and to actively participate in class discussions with their peers.

Motivating students to study before class has been a challenge in other disciplines and one of the more promising solutions that has emerged is the concept of *peer instruction*. This teaching paradigm was introduced by Eric Mazur, a physicist at Harvard University, who realized that his students could pass their traditional, formulaic problem-solving exam but have little conceptual understanding of Newtonian physics [2], [11]. When confronted with new types of questions on the same concepts, they were simply not able to adapt.

In a peer instruction classroom, lecture is interspersed with multiple-choice questions known as ConcepTests, which are designed to provoke deep conceptual thinking in students and engage them in meaningful discussion with their peers. Peer instruction has been shown to improve outcomes in several scientific disciplines, such as physics, computer science, and biology. In computer science (CS), it has shown promising results such as halving failure rates in four different courses [4] and increasing retention in the major [8].

Although reports from the field show the successes of peer instruction in CS, the current focus has been limited to theoretical and introductory programming courses [3]. In our experience, there are substantial differences between teaching a standard CS course and an advanced cybersecurity one. For instance, cybersecurity teaching is expected to transform students' mindsets for increased adaptability and analytical skills for assessing dynamic risks and defense strategies. Unfortunately, peer instruction is not widely used in cybersecurity education and a major challenge in adopting peer instruction is the development of in-class, conceptual questions appropriate to the unique goals of cybersecurity education.

Over the past ten months, the authors have made significant efforts and developed 172 peer instruction questions for two cybersecurity courses: introduction of computer security, and network penetration testing. This paper analyzes these questions and identifies a systematic methodology for developing peer instruction questions. Furthermore, it provides four examples of the peer instruction questions in cybersecurity.

The rest of the discussion is organized as follows: Section §2 provides some background for peer instruction. Section §3 presents a methodology for developing peer instruction questions, and section §4 provides examples of peer instruction questions for four major cybersecurity areas. Section §5 provides a comprehensive analysis of the peer instruction questions recently developed by the authors for two cybersecurity courses. Finally, section §6 concludes the discussion.

2 Peer Instruction Background

2.1 Peer Instruction Methodology

The peer instruction method divides the lecture period into small presentations. Each presentation focuses on a central point and is typically followed by a series of the following activities:

- A conceptual question is asked to students, who are then given two to three minutes to formulate individual answers and report them to the instructor. Typically, the question is in multiple-choice format, enabling aggregation of student response data by the instructor.
- If a mix of correct and incorrect answers is received, students are further encouraged to discuss their answers with others sitting around them. The discussion may last three to four minutes. The goal of the discussion is to present the fundamental reasons behind the answers and for students to convince each other of the correctness of their own answers.
- Students are then asked to stop discussion and polls for their answers are performed again to observe how their opinions were influenced by the discussion in the previous step.
- After reviewing the poll results, the instructor decides to either move on to the next concept or present the correct solution with more explanation, as needed.

Peer instruction requires students to be better prepared for each class. The instructor provides reading material on the topic to be covered in the class and the students have to read the material before the class, allowing them to better understand the presentations and respond to conceptual questions. The students are also given a quiz to solve after reading the material, and some incentives (such as bonus marks) are associated with each quiz. This approach encourages students to go through the material carefully and to be prepared.

2.2 Peer Instruction Outcomes in CS

More recently, peer instruction has been introduced in computer science, and research has shown that computer science students both value peer instruction and also recommend that more instructors use it at both small colleges and large schools [12], [7]. Research also shows that instructors who use a peer instruction approach in their classrooms find it quite effective [5]. Essentially, the real learning occurs during discussions between students when a conceptual question is asked to them [6].

Research also shows that students who have learned through peer instruction achieve 6% higher grades on their final exams than students in a lecture-centric standard teaching environments [13]. Peer instruction has shown effective results in reducing failure rates by 61% on average in four computer science courses (CS1, CS1.5, Theory of Computation, and Computer Architecture) [4]. It has also shown a 31% improvement in the retention of students in a computer science major [9].

3 Question Development Methodology for Peer Instruction

3.1 Challenges for Developing Questions

In our experience, there are a number of challenges that arise when developing peer instruction questions. In particular, this section discusses two challenges that are encountered frequently by the authors.

Quiz vs. Peer Instruction Questions. The main challenge is the development of multiple-choice conceptual

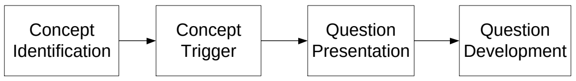


Figure 1: Overview of the methodology for developing peer instruction questions

questions to facilitate peer discussion. Questions reserved for quizzes rather than peer instruction need to be less conceptual, as they are typically used to test knowledge rather than the application of knowledge. It can be difficult to step away from that for conceptual questions without these questions seeming too simple.

Plausible Incorrect Answers for a Peer Instruction Question. Similarly, there is a difficulty in creating incorrect answers for peer instruction questions that seem plausible. The "trolling for misconceptions" and similar question tactics by Beatty et al. [1] assist in fulfilling this issue, though it is occasionally also difficult to realize potential misconceptions.

3.2 Overview of the Methodology

We have identified a basic methodology for creating peer instruction questions systematically. Figure 1 illustrates four stages of the methodology: concept identification, concept trigger, question presentation, and question development. To develop a peer instruction question, the first stage is to identify a concept defining the main focus of a question, and then further (in the second stage) identify a concept trigger that is introduced in the question to provoke a student's thinking process and set the desired direction of peer discussion. Deliberately introducing ambiguity in answer choices is an example of concept triggers (more examples are discussed in Table 1). Multiple concept triggers can be used for a question.

The third stage determines how the concept and the concept-trigger(s) identified in last two stages can be put together in a question for better presentation, and easier understanding. For instance, a question can be presented in a scenario, example, or diagram. The last stage of the methodology creates the question, including articulation and identification of multiple choices.

4 Examples of Peer Instruction Questions

This section presents four examples of peer instruction questions representing four distinct cybersecurity areas: introductory cybersecurity concepts, digital forensics, reverse engineering, and network penetration testing. The first is typically taught in the traditional lecture format, whereas the latter three are intensive hands-on courses. Together, they form the basis of a broad skill set and perspective on security.

The section further discusses concept triggers and question presentation for each question. We borrow our concept triggers from Beatty et al. [1], and used them for the detailed analysis of our peer instruction questions (discussed in §5). Concept triggers (mostly used in our analysis) are briefly described in Table 1. Furthermore, we identify five question-presentation types after carefully analyzing our peer instruction questions: Scenario, Examples, Definitional, Diagram, and Feature.

Scenario questions present students with a situation, and require students to answer the question provided about the situation by examining the literal and implied details of that situation. *Example* questions simply provide or describe a sample system or code—these are somewhat similar to scenario questions, but are more straightforward, as there is less interpretation required for students to understand and respond to the question. *Definitional* questions are even simpler—they deal strictly with the definition of a concept, and are used best when attempting to differentiate between two or more particularly similar concepts. *Diagram* questions present students with a diagram and ask them to make interpretations based on the visual—these must not simply be code snippets; they must have a strong visual component. Finally, *feature* questions deal with the components of a concept—they are questions that may ask whether a provided example has all of the required features of that concept, which feature is a major component of the concept, or which concept the provided features best support.

4.1 Introductory Cybersecurity Concepts

This example of a peer instruction question is used to introduce the key concepts of confidentiality, integrity, and availability (CIA triad). A pre-class reading assignment for students includes basic overview of the CIA triad. In class, the instructor initially presents some slides and explains the concepts, and then shows the peer instruction question. Students are given an opportunity to answer individually first. After all students have responded, students discuss their answers in small groups and come to a consensus. This approach gives students the opportunity to engage in discussion and problem solving with their peers to arrive at an answer.

One example of the peer instruction question could be: *An attacker deletes files on a system, denying access to system users. Which element of CIA triad is violated?* a) Confidentiality, b) Integrity, c) Availability, and d) None/Other/More than one of the above.

Concept trigger. We can deconstruct this peer instruction question and identify some of the question design tactics by Beatty et al. that might be used to construct a question such as this one. First, by noting the question option D, this question introduces the usage of "none of the above" as well as "identify a set or subset", by allow-

Table 1: Sample concept-triggers borrowed from Beatty et al. [1]. We use them for analyzing peer instruction questions

Concept triggers	**Description**
Compare and contrast	Compare multiple situations; draw conclusions from comparison
Interpret representations	A situation that asks students to make inferences based upon the presented features
Identify a set or subset	Ask them to identify a set or subset fulfilling some criterion
Strategize only	Provide a problem; ask students to identify the best means of reaching a solution
Omit necessary information	Provide less information than is essential for answer; see if students realize this
Use “none of the above”	Provide an option to learn alternative understandings
Qualitative questions	Focus on concepts and relationships rather than numbers or equations
Analysis and reasoning questions	Require significant decision-making, hence promote significant discussion
Trap unjustified assumptions	Choices are facilitated by potential unjustified assumptions made by the students
Deliberate ambiguity	Use deliberate ambiguity in questions to facilitate discussion
Trolling for misconceptions	Trapping students with answers that require common misconceptions to choose
Remove nonessentials	Strip the question down; remove potential distractions
Extend the context	Ask familiar question of a new, unknown situation
Reuse familiar question situations	Ask a different question of a known situation; save students’ cognitive resources
Oops-go-back	Ask a pair of questions; first question is designed to trap students with a common error; the second clarifies the situation and helps students realize the error
Rank variants	Similar to identify a set or subset, but “variants are ranked according to quality”
Reveal a better way	Present a problem that students will likely solve with a difficult or convoluted solution, and then suggest a simpler solution during discussion
Include extraneous information	Ask a question that requires students to identify exactly what is relevant in the question to answer the question correctly
Answer choices reveal likely difficulties	Provide question answers that highlight misunderstandings and common errors
Multiple defensible answers	Provide questions with multiple viable answers depending on interpretation
Require unstated assumptions	Ask a question that requires particular assumptions not evident in the question; this potentially leads to “multiple defensible answers” as well

ing the students to select none of the above or any option that they may choose from the set.

Additionally, this question is qualitative rather than quantitative as it approaches the concept of the balance of the CIA (Confidentiality, Integrity, and Availability) triad and its components, representing the tactic of “qualitative questions”.

Finally, this question promotes the usage of “multiple defensible answers”, as while it lends itself primarily to exhibiting a violation of availability (due to the phrase “denying access to system users”), this attack would secondarily be considered a violation of system integrity, and as such, choice C would be the primary answer, but choice B is also acceptable. Ultimately, the answer is D.

Question presentation. This question is presented as an example. However, it can be elaborated in a scenario.

4.2 Digital Forensics

In *digital forensics*, the problem of comprehensive data recovery is a critical first step for most inquiries. In most cases, the formatting of a hard disk has a small impact on the actual data content, because it simply overwrites file system metadata (making the data inaccessible via the OS interface). There is a common misconception, even among users with a strong technical background, that format operations permanently destroy file content. In fact, with the exception of internal SSDs, most format operations destroy very little file content. One example question to make students think about it, could be: *Estimate the fraction of disk blocks affected by formatting the disk:* a) 100%, b) 65%, c) 20%, d) Less than 5% (answer: d). Follow-up questions could lead them to the fact that the answer may be different, depending on the target medium—HDD vs. SSD vs. virtual disk.

The corresponding peer instruction lesson involves describing basic filesystem layouts, then asking students to take a position on what percentage of filesystem data would be destroyed by first formatting a USB flash drive using the FAT filesystem. The scenario is then expanded to include a sequence of other format operations using different filesystem types, including NTFS, ext3 on Linux, etc., against the same flash drive. Students are asked to agree how to visually depict how much data they predict is permanently destroyed. The instructor then uses a visual aid to illustrate exactly how much data is destroyed. The results are illustrated in Figure 2. Figure 2 illustrates data deletion (in red) as the flash drive is subjected to a number of format operations. Green areas remain untouched by format operations. As a pre-class reading assignment, the students cover the reading material that would approach the subject of data deletion vs. data destruction.

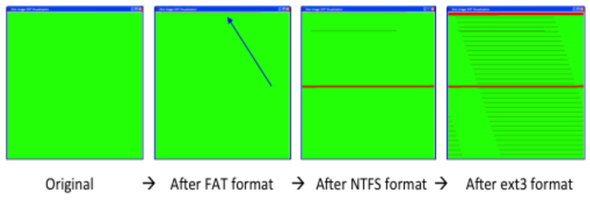


Figure 2: The results of formatting a flash drive with the FAT, then NTFS, then the EXT3 filesystem. Deleted regions are shown in red.

```

Start:
    mov word   ptr loc_10106+1, 152h
loc_10106:   ;DATA XREF:
    mov ax, 168h
    mov word ptr loc_10129+5, ax
loc_10129:   ;DATA
    mov word ptr es:0, 4D4Ch
```

Listing 1: Self-modifying code snippet

Concept trigger. Deconstructing this question, it uses some interesting question tactics. First, formatting a disk can have vastly different effects depending on the filesystem type—this is a particularly important detail missing. Initially, this question can be noted as using "omit necessary information" as well as "trap unjustified assumptions", as any particular answer would need to assume a particular formatting scheme—"trolling for misconceptions" could work as well, if students assume that this operation is performed the same regardless of filesystem type. Finally, this question involves "deliberate ambiguity", and it is a "qualitative question". It is important to understand the power of this question's design tactics: leading a discussion or even a curriculum by confronting learners with their own misconceptions heightens the impact of the subsequent correct information reveal [10].

Question presentation. This question could be identified primarily by *feature*, as it requires an understanding of disk formatting, and students will need to understand the features of that process to guess the correct percentage of blocks affected—even though this question intentionally leaves out a significant amount of information.

4.3 Reverse Engineering

In *reverse engineering*, it is important not only to understand assembly-level language, but it is also critical to know the specifics of its implementation on particular hardware. With assembly language, the programmer has fewer restrictions in terms of access to memory than in higher-level languages such as Java. In effect, this allows the programmer to easily create self-modifying code that overwrites other instructions or operands in memory. However, instruction prefetch caching introduces additional complexity. If code is modified in memory, the modification will persist, but if prefetch is enabled, this only applies to current control flow to an extent. In the case of a small number of instructions (16 bytes for the Intel 80486 prefetch queue), if an instruction modifies code that has previously been fetched, it will be executed as if it were not modified in memory. If prefetch caching is not enabled (as is the case when single-stepping in a debugger), modified code that falls in control flow will be executed as if it is modified, regardless of locality to the instruction pointer. An example question to explore this concept is, given a code example (shown in Listing 1) that presents code modification within prefetch range of control flow: *After executing these instructions while single stepping inside a debugger on an* 80486 *processor, what is the value of the 16-bit word at location* loc_10129+5*?* a) 168h, b) 152h, c) 4D4Ch, d) Value is unknown, e) None of the above.

The corresponding discussion first explores prefetch caches, as well as explaining more modern uses of the cache (ex: clearing the cache when a branch occurs). It also explains how a debugger handles self-modifying code while single stepping (in the context of clearing the cache on each step), and then uses a portion of the Intel IA-32 manual to explain how certain families handle self-modifying code. The peer instruction question can then be asked. As a pre-class reading assignment, the students cover a basic overview of anti reverse engineering techniques such as encryption, and anti-debugging.

Concept trigger. Deconstructing this question, it allows for the usage of "none of the above". Additionally, this question requires quite a bit of analysis—students must read and understand the code snippet as well as what the question is asking in order to better understand how it will ultimately execute in the particular situation—so this question falls under the category of "analysis and reasoning questions".

Question presentation. This question is clearly an example question, as it provides a code sample and requires that students answer by making inferences from the code sample as well as the question itself.

4.4 Network Penetration Testing

The problem of obtaining user credentials is central to most aspects of network penetration testing. Therefore, a non-trivial amount of time is spent on various password cracking techniques. The real point is, of course, is to gain an understanding of what methods can be used to thwart such attacks. A common point of misunderstanding is the purpose of salting password hashes, and what types of problems it can address. Salting is the prepending of random bits to the password prior to hashing.

Although the salt is not a secret, it can render efforts to reverse password hashes that rely on precomputed mappings computationally infeasible. At the same time, salting does little to prevent the cracking of weak passwords, as they can be effectively broken with dictionary techniques. An appropriate conceptual question that can lead to the various considerations is: *You obtain a leaked database of unsalted SHA-1 password hashes. What would be the most effective way to obtain as many passwords as possible in a short amount of time?* a) brute force, b) rainbow tables, c) dictionary attack with a large wordlist, d) passing the hash, and e) birthday attack.

The corresponding lesson first begins by discussing simple password guessing, noting that simple guessing is limited by speed as well as account lockouts. Then it moves to means of automation, pointing out that automation works best against collections of hashes, allowing for much greater speeds. Methods of password cracking such as rainbow tables and dictionary attacks can be discussed, leading to the conceptual peer instruction question to gauge understanding of the situational advantages. Following the question and discussion, the lecture can then turn to means of optimization of these approaches through heuristics, and then, the instructor can demonstrate common password cracking tools as well as provide an example hash for a hands-on lab. As a preclass reading assignment, the students cover a basic concept of password hashing and salting.

Concept trigger. Deconstructing this question, it is "qualitative", as it provides enough information about the password hashes to require students to identify concepts (hashing, salting, etc.) and understand the relationship between the presence or stated lack (hash salting) of concepts to correctly answer the question, rather than a simple equation. Secondly, this requires students to "interpret representations"—they must identify the key words and concepts and make interpretations based on their usage. Finally, this question uses the "strategize only" question trigger, as students must identify the best path or tool to a solution rather than a solution itself.

Question presentation. This question is clearly a scenario question, as the question presents a situation (acquisition of a hash dump).

5 Analysis of Peer Instruction Questions

The authors have developed 172 peer instruction questions for two cybersecurity courses: introduction to computer security (93 questions), and network penetration testing (79 questions). The paper cannot contain all the questions due to limited space. Thus, to provide some insight into the questions in the context of the question development methodology (refer to §3), this section presents an analysis of the questions. The goal of the analysis is to identify the concept triggers (in Table 1) and presentation types in the questions of both courses, and then, compare them.

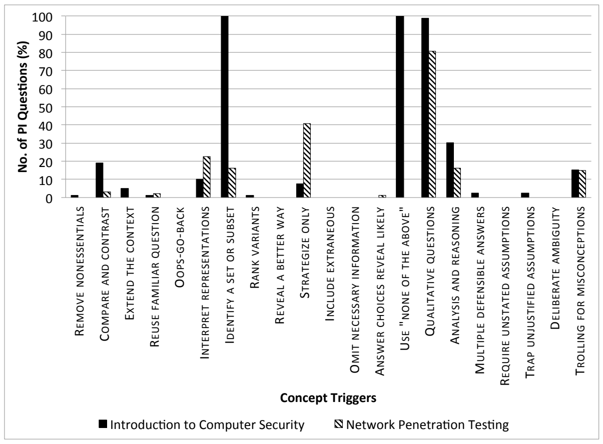


Figure 3: Percentage of peer instruction questions over concept triggers

For introduction to computer security, all 93 questions have multiple triggers. Excluding the "none of the above" trigger (as this is present in nearly all the questions due to the none/more than one of the above option), we have 84 questions with multiple triggers. Excluding the "identify a set" trigger (as this is also present in nearly all the questions due to the same option), we have 83. Excluding both of those concept triggers, we have 56 questions with multiple triggers. For network penetration testing, 78 out of the 79 have multiple triggers, due to questions with the qualitative question trigger.

5.1 Concept Trigger

Figure 3 presents the percentage of questions having concept triggers under consideration. There are a number of concept triggers conspicuously missing from both courses. This is because when building a question with particular concept triggers, there are many (such as "remove nonessentials") triggers that work well when developing a question, but are difficult to identify when dissecting a conceptual question; however, a number of concept triggers present in questions are in fact evident, and provide an interesting view into the creation of a peer instruction question set as a whole. While it may be difficult to identify items such as oops-go-back pairs, the ability to view trends of concept trigger usage shows interesting insight into the intentions of the instructor.

Both courses have a majority of questions that are qualitative. It can be rationalized, as the peer instruction questions are generally qualitative because they refer to concepts, relationships between concepts, etc.

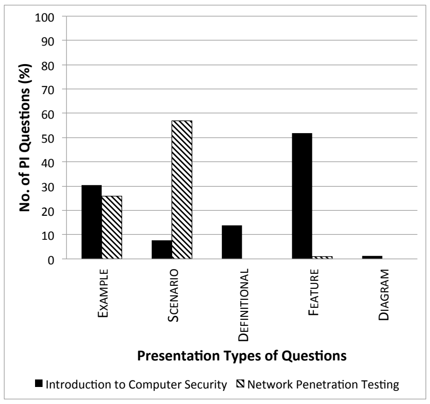


Figure 4: Percentage of peer instruction questions over presentation types

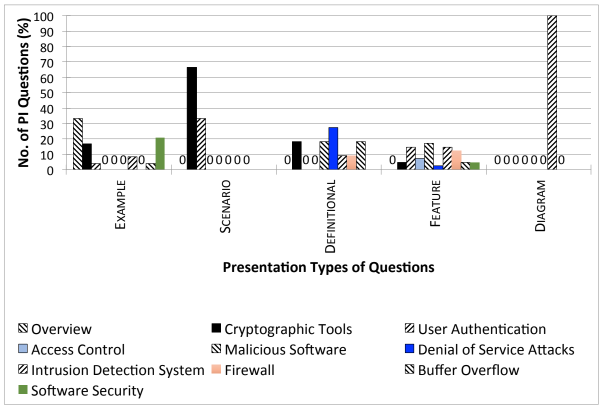


Figure 5: Percent distribution of questions as per presentation types and topics

5.2 Question Presentation

We categorize the peer instruction questions based on the wording or presentation of the questions. The presentation types identified in our questions are scenario, example, definitional, diagram, and feature.

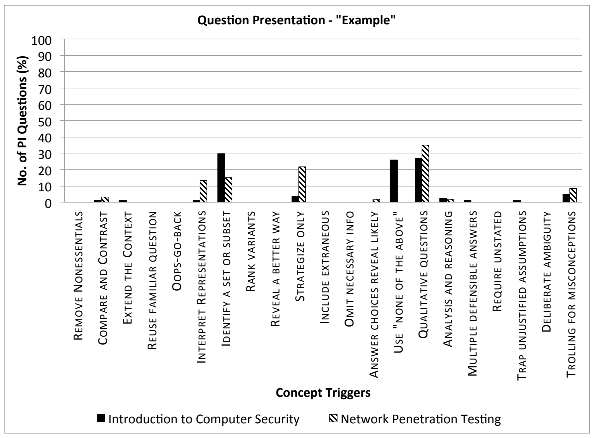
Figure 4 presents the percentage of peer instruction questions over presentation types. Both of the courses have a similar number of example questions. However, the introduction to computer security course has a significant number of feature questions. The network penetration testing course, on the other hand, has a majority of scenario-based questions.

There is no strict rule as to when to use each question presentation type; this is up to the instructor. However, it is important to consider the class material. For example, in our penetration-testing course, most of the material discussed lends itself well to hands-on activities, and much of the class time is used for hands on activities in a lab. For concepts that arise in practical material such as this, it would be helpful to trend toward using more scenario-based questions. When discussing subjects such as relationships between concepts or objects—for example, redirection of *stdin* and/or *stdout* through a *Netcat* instance—an instructor may find it useful to provide a diagram and focus the question around that. It is largely up to the instructor, but question presentation, much like question triggers, should be used to enhance peer discussion as deemed necessary.

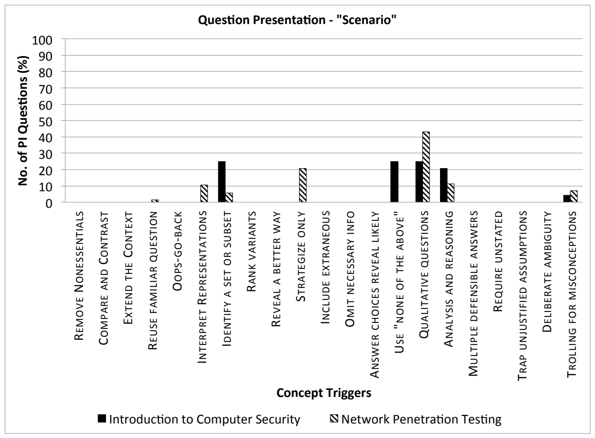
We further analyze Figure 4 data to answer two questions: 1) what presentation types are used more frequently for different cybersecurity topics, and 2) what is the association between presentation types and concept triggers. Both aspects are critical for developing an effective peer instruction question.

Presentation types vs. cybersecurity topics. To answer the first question, we have only considered the peer instruction questions for the course introduction to computer security. The course covers ten significantly different areas of cybersecurity (listed in Figure 5), and its questions are spread out across all the presentation types (refer to Figure 4). For the analysis, we further find the distribution of questions in accordance with cybersecurity topics and presentation types. Figure 5 presents the results showing that scenario based questions are from the topics, cryptographic tools, and user authentication. Apparently, these topics are traditionally discussed in scenario settings such as exchange of shared keys by Alice and Bob, or an attack scenario to steal and brute force a password file. Example, definitional, and feature based questions are spread out across the topics. Interestingly, the whole dataset contains only one diagram question, and that is for buffer overflow.

Association between presentation types and concept triggers. Figure 6 (and Figures 7 and 8 in Appendix) presents the results of the distribution of questions of a certain presentation type with respect to concept triggers. The analysis of the results shows that both courses utilize similar concept triggers for Example and Scenario-based questions. The exceptions are "none of the above", and "strategize only" that are only used by for the introduction to computer security course and network penetration course, respectively. Definitional and feature types of questions are mostly used for the introduction to computer security course. Interestingly, "Compare and contrast" is used in both courses, but for different presentation types, i.e., definitional, and feature. Presentation type "Diagram" only has one question in our dataset for the course, introduction to computer security, utilizing three concept triggers: "Interpret Representations", "Identify a set or subset", and "none of the above". In general, Diagram questions are time-consuming and more difficult to create,and are therefore less likely to be popular for peer instruction questions.



(a) Questions of presentation type "Example" are analyzed



(b) Questions of presentation type "Scenario" are analyzed

Figure 6: Association between concept triggers and question presentations (such as example, and scenario).

6 Conclusion

Through the use of peer instruction, we seek to build problem solving skills and technical aptitude in students who take advanced cybersecurity coursework. The expectation of student preparation prior to class and significant discussion during class significantly help students to have better thingsstanding of the content and better learning experience in class.

Our question development methodology for peer instruction allows instructors to systematically create questions and smoothly transition from lecture style format to peer instruction. The results of our analysis of 172 peer instruction questions (developed for two cybersecurity courses) conclude that the example and scenario based questions are more suitable for peer instruction questions. The concept trigger "qualitative question" generally applies to peer instruction questions. However, depending on the subject area in cybersecurity, the concept triggers may or may not be appropriate for the peer instruction questions. For instance, concept triggers "identify a set or subset" and "strategize only" are mostly suitable for the introduction to computer security course and network penetration-testing course, respectively.

As part of the future work, we plan to utilize the peer instruction questions in their respective courses, and evaluate their overall efficacy in class.

References

[1] Beatty, I., Gerace, W., Leonard, W., and Dufresne, R. Designing effective questions for classroom response system teaching. *American Association of Physics Teachers 74*, 1 (2006).

[2] Crouch, C. H., and Mazur, E. Peer instruction: Ten years of experience and results. *American Journal of Physics 69* (2001).

[3] Lee, C., Garcia, S., and Porter, L. Can peer instruction be effective in upper-division computer science courses. *ACM Transactions on Computing Education 13*, 3 (2013).

[4] Porter, L., Bailey-Lee, C., and Simon, B. Halving fail rates using peer instruction: a study of four computer science courses. In *Proceedings of the 44th ACM technical symposium on Computer science education* (Denver, CO, March 2013).

[5] Porter, L., Bailey-Lee, C., Simon, B., Cutts, Q., and Zingaro, D. Experience report: a multi-classroom report on the value of peer instruction. In *Proceedings of the 16th Annual Conference on Innovation and Technology in Computer Science Education* (Darmstadt, Germany, June 2011).

[6] Porter, L., Bailey-Lee, C., Simon, B., Cutts, Q., and Zingaro, D. Peer instruction: do students really learn from peer discussion. In *Proceedings of the 7th Annual International Computing Education Research Workshop* (Providence, RI, August 2011).

[7] Porter, L., Garcia, S., Matusiewicz, J. G. A., and Taylor, C. Peer instruction in computer science at small liberal arts colleges. In *Proceedings of the 18th Annual Conference on Innovation and Technology in Computer Science Education* (Canterbury, England, July 2013).

[8] Porter, L., and Simon, B. Retaining 18-30% more majors with a trio of instructional best practices in cs1. In *Proceedings of the 44th ACM technical symposium on Computer science education* (Denver, CO, March 2013).

[9] Porter, L., and Simon, B. Retaining nearly one-third more majors with a trio of instructional best practices in cs1. In *Proceedings of the the Special Interest Group on Computer Science Education Technical Symposium* (2013).

[10] Schank, R. C. *Dynamic Memory Revisited*. Cambridge University Press, 1999.

[11] Simon, B., and Cutts, Q. Peer instruction: a teaching method to foster deep understanding. *Communications of the ACM 55*, 2 (2012).

[12] Simon, B., Kohanfars, M., Lee, J., Tamayo, K., and Cutts, Q. Experience report: peer instruction in introductory computing. In *Proceedings of the 41st SIGCSE technical symposium on computer science education* (Milwaukee, WI, March 2010).

[13] Simon, B., Parris, J., and Spacco, J. How we teach impacts student learning: peer instruction vs. lecture in cs0. In *Proceedings of the 44th ACM technical symposium on Computer science education* (Denver, CO, March 2013).

A Appendix

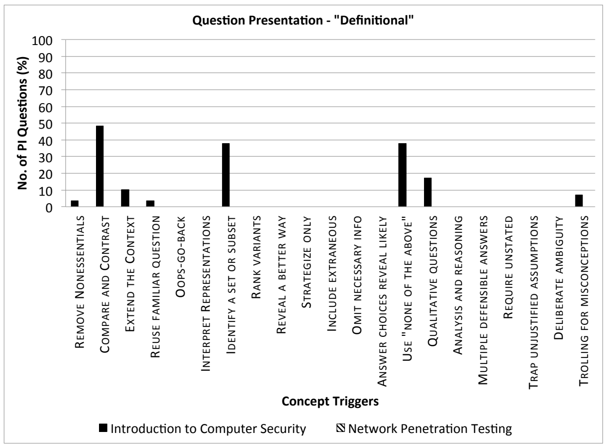


Figure 7: Questions of presentation type “Definitional” are analyzed

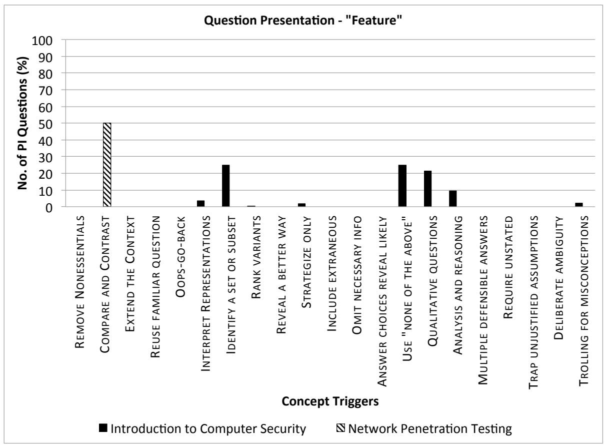


Figure 8: Questions of presentation type “Feature” are analyzed