Choose Your Pwn Adventure: Adding Competition and Storytelling to an Introductory Cybersecurity Course

Tom Chothia, Chris Novakovic, Andreea-Ina Radu, and Richard J. Thomas

School of Computer Science
University of Birmingham, UK
{t.p.chothia,c.novakovic,a.i.radu,r.j.thomas}@cs.bham.ac.uk

Abstract. Narrative is an important element of gamification. In this article, we describe the development of a framework that adds a narrative to an 11-week cybersecurity course. The students play the part of a new IT security employee at a company and are asked to complete a number of security tasks, for which they receive flags. As well as being used to assess their performance throughout the course, students can send the flags they find to a number of different characters to progress the storyline in different ways. As the story unfolds they find deceit, corruption and ultimately murder, and their choices lead them to one of three different endings. Our framework for running the story and the exercises is completely self-contained in a single virtual machine, which the students each download at the start of the course; this means that no resource-consuming backend or cloud support is required. We report on the results of qualitative and quantitative evaluations of the course that provide evidence that both the VM and the story contained within it increased student engagement and improved their course results.

1 Introduction

Live security exercises are a popular and fun means of engaging with cybersecurity topics in academia, industry [8], and government [21] alike. The pedagogical benefits of these exercises have been widely reported (e.g., [29,19,9,18,1,12]); participants often cite the enjoyment and satisfaction of competing, their increased motivation to learn about cybersecurity, improved practical knowledge of theoretical aspects of cybersecurity, and the discovery of deficiencies in their knowledge as benefits of taking part.

One common type of live security exercise is the *Capture The Flag* (CTF) competition. The objective is for participants (whether individuals or teams) to defend a host running vulnerable services while simultaneously attacking other teams' hosts running the same services (an *attack/defence-style CTF*). Alternatively, participants can solve standalone challenges from a range of categories, including binary reverse-engineering, forensics, and web security, without the need to interact directly with the other participants (a *Jeopardy-style CTF*).

The successful exploitation of a vulnerable service or solving of a challenge reveals a secret piece of information known as a *flag*, which scores points for the participant when submitted to a *flag server* operated by the CTF organisers; the submission of flags for more difficult challenges scores a greater number of points for the participant, and the participant that scores the most points over the course of the competition is declared the winner. Some attack/defence-style CTFs also incorporate elements of Jeopardy-style CTFs. In both cases, the exercise typically runs over a short period of time (usually 12–48 hours).

Live security exercises commonly have themes or simple linear narratives to give structure to the exercise, connect otherwise-disparate technical challenges, and to amuse or sustain the interest of their participants. The long-running academic CTF, iCTF, has featured a variety of storylines since it began in 2001, from infiltrating and ultimately disarming a terrorist network (with the aim of defeating the terrorists more quickly than the other participants) to running a botnet-based organised crime operation (with the aim of amassing more illicit money from victims than the other participants by the end of the competition) [29]. The 2014 edition of the high-school CTF picoCTF featured a storyline about helping a broken robot to get home [5], while the 2017 edition required participants to locate and rescue a friend who suddenly and unexpectedly disappears [4]. Cybersecurity camps, elaborate exercises requiring participants to be physically present in a given location over a longer period, afford their organisers the ability to form more detailed and immersive narratives than CTFs: Feng et al. developed a camp with a storyline based on the theme of the *Divergent* young-adult science fiction novel series [11], while the industry-led HackFu event hires professional actors to play characters who interact with participants to progress the storyline [20]. Similarly, the theme for the Cambridge-led cybersecurity camps Inter-ACE [28] and Cambridge 2 Cambridge (C2C) [17] 2017 was cybersecurity warfare, placing participants in the scenario of having to defend the nation against a rogue organisation. Participants had to discover and take down strategic assets belonging to adversaries, as well as search for forensic clues on hard drives recovered from behind enemy lines.

Given the pedagogical benefits of live security exercises, instructors may find it desirable to introduce them into the curricula of academic cybersecurity courses. However, there are several potential barriers to doing so.

The funding to run more immersive forms of live security exercise, such as cybersecurity camps, is rarely available for individual academic courses; with their lower operating costs, CTFs are more favourable for inclusion in academic curricula. Even so, CTFs require continuous supervision by their organisers to guarantee the smooth running of the competition; a pool of organisers working intensive shifts can ensure that a 48-hour CTF runs successfully, but this is not viable in an academic teaching scenario where courses run over a period of several months.

There are also technical barriers: the infrastructure underpinning a CTF is typically large and complex (e.g., [29,9]), and the computing power (and, for attack/defence-style CTFs, network bandwidth) required for the smooth operation of a CTF is usually not available for teaching purposes. Additionally, CTFs inevitably involve attacking vulnerable network services, and there is a danger that students' malicious network traffic could interfere with other network hosts that are not part of the competition; we (and others [3]) have found university IT support staff hesitant to allow malicious traffic to pass over any network they control.

We have developed a framework for including Jeopardy-style CTF challenges in academic cybersecurity courses, and have integrated it into our own 11-week undergraduate course. The framework takes the form of a virtual machine (VM) containing vulnerable services and challenges devised by the course staff; they can be revealed gradually to students as the relevant cybersecurity topics are taught in lectures, allowing individual exercises to feature particular challenges. Each student runs the VM locally and attempts to solve each challenge inside the VM as it is made available by the course staff. The successful completion of a challenge reveals a flag to the student, which can be submitted to a flag server controlled by the course staff for credit in the exercise; flags are unique to a particular instance of the VM, allowing for the detection of collusion between students.

The framework also features a *narrative engine* with which students may optionally engage with: as well as gaining course credit by submitting discovered flags to the flag server, students may progress a non-linear storyline by emailing the same flags to fictional characters that exist within the VM. The narrative engine is reusable: the narrative is contained in a single file (the *story map*) that defines the identities of the characters, the story that connects them (and the student), and decisions that may be taken by students at particular points during the story to influence the plot. The narrative we have developed for our course follows screen-writing best practice (e.g., [27]), and is designed to be compelling enough that students become attached to the characters and therefore feel invested in progressing the storyline to completion over an 11-week period.

Our framework provides the benefits of live security exercises to cybersecurity students while avoiding the drawbacks discussed earlier: students run the VM on their own hardware (e.g., their laptops), so course staff need not invest time continually maintaining a centralised infrastructure to operate the CTF, and malicious traffic is only routed inside a particular student's virtual network, eradicating the impact that an inexperienced student's actions may have on the university network. Since the framework is reusable, instructors can easily make changes to both the challenges and narrative between iterations of the course (e.g., in order to incorporate student feedback gathered during previous iterations).

Our framework has been used in some form in all four iterations of our course. The framework has driven all of the continuous assessment in each iteration; we therefore expected students to interact with it for 3–4 hours per week over 11 weeks. The use of VM-based CTF-style exercises was popular with students, as indicated by high student satisfaction levels reported in the end-of-course anonymous feedback questionnaires. A storyline was added in time for the fifth iteration of the course, where we found that following the storyline led to improved student attainment and overall student engagement, as well as improved development of students' knowledge of cybersecurity issues.

The rest of this article is organised as follows. We provide a brief overview of the structure and content of our undergraduate cybersecurity course at the University of Birmingham in Section 2; our own requirements influenced design decisions we made while creating the framework, although it is flexible enough that instructors for cybersecurity courses at other universities can provide their own exercises and narrative within the same framework. We then describe the structure of the framework itself: Section 3 describes the architecture of the VM distributed to students at the start of the course, and Sections 4 and 5 describe the infrastructure that controls progression of the story that takes place within the VM. Section 6 outlines the story we developed for our course and the decisions that students can make in order to influence the plot. In Section 7, we provide an evaluation of the framework and story based on an analysis of students' marks for several iterations of our course and feedback we gathered from them after each iteration. We conclude in Section 8.

A website with additional information, a downloadable copy of our latest VM, and sample exercise sheets can be found at `https://www.cs.bham.ac.uk/internal/courses/comp-sec/vm_story`. Parts of the current work have previously been published in [7,6].

2 Our Cybersecurity Course

Since the 2013/14 academic year, the School of Computer Science at the University of Birmingham has offered an optional introductory cybersecurity course to second-year undergraduates, with a particular focus on technical skills and understanding. The course runs for 11 weeks, with 2 hours of lectures and 2 hours of lab sessions per week. It is assessed both formatively (via continuous assessment, worth 20% of a student's final grade) and summatively (via exam, worth the remaining 80%). Topics covered in the course have included cryptography, access control, network and protocol security, web security, buffer overflows, and reverse-engineering of binaries. Students enrolling on the course are expected to arrive with basic programming and networking skills.

Prior to the 2015/16 academic year, the School also offered a graduate-level introductory cybersecurity course. While this course covered many of the same topics as the undergraduate-level course, and also integrated the framework we present in this article into its continuous assessment (with many of the same outcomes), in this article we focus exclusively on the undergraduate-level version of the course. We encourage the reader to refer to our earlier work [7] for an analysis of the graduate-level course.

The continuous assessment for the course is delivered in the form of exercises, lasting for 2–3 weeks each, that are intended to test students' in-depth understanding of the topics covered in lectures; the difficulty of the exercises is set based on experience gained from teaching previous courses and knowledge of other courses that the students have taken.

2.1 Marking and Grading Criteria

Each exercise is marked out of 100; grade boundaries are positioned at intervals of 10 marks between 40 and 70, with a mark above 70 representing high-quality work and a mark below 40 representing failure. Generally, a third of the marks are awarded for a basic solution (e.g., for the protocol security exercise, a working attack against the protocol), another third are awarded for an optimal solution (e.g., a protocol attack with no unnecessary steps), and the final third are awarded for showing a clear understanding of the problem (e.g., a description of each line of the protocol attack, making it clear what each part of the message does, and why the attack works). Students were informed in advance of how marking would take place and were told what was expected of a high-scoring submission, but did not expect to receive top marks for all of their continuous assessment. Marks were not scaled; as shown in Section 7, they naturally ended up distributed across all grades.

For all exercises, students were required to submit written answers describing the steps they took to recover flags from the VM, and — where appropriate — a description of what the vulnerabilities were and how they worked, and an explanation of how they could be fixed. Not all questions required the recovery of flags from the VM, and flags were not used to unilaterally prove that the student completed the exercise, but were instead intended to give the marker some degree of assurance that the students' written answers were dependable. There were minor variations to the questions in the exercises for each iteration of the course, to prevent students from simply reusing solutions from a previous iteration. Examples of the full exercise sheets are available from `https://www.cs.bham.ac.uk/internal/courses/comp-sec/vm_story`.

2.2 Exercise Content

Each iteration of the course has featured continuous assessment consisting of either four or five exercises corresponding to the topics taught in lectures. We now describe how Jeopardy CTF-style challenges are integrated into each exercise, and outline how students are expected to complete the challenges in order to find them.

Exercise 1: Basic encryption. The first exercise familiarises students with performing cryptographic operations in Java. The students are given the passwords to two user accounts on the VM, `alice` and `bob`. `bob`'s home directory contains incomplete Java source code for an encryption application; the supplied code contains methods for encrypting files using AES (in either CTR or CCM mode) and RSA using a user-defined key. The CTF-style challenges for this exercise involve decrypting AES/RSA-encrypted files in `bob`'s home directory that each contain a flag; students are required to write the corresponding decryption methods for the application and use their compiled code to recover the flags from the encrypted files. Additionally — as an example of a question that is not assessed as part of a CTF-style challenge — a file in `bob`'s home directory contains the message "Pay Tom 1000 pounds" encrypted using AES in CTR mode; students are not given the encryption key, and must therefore manually edit the bits of the ciphertext in the file so that, when decrypted using the code they have written, the corresponding plaintext reads "Pay Bob 9999 pounds".

Exercise 2: Access control. The VM contains a number of files protected with flawed access controls. One flag is stored in a file hidden in an obscure location in the file system rather than being given appropriately restrictive permissions; others are stored in files with a variety of insecure permissions that allow the file to be read (e.g., via a chained confused deputy attack against two insecure programs with the `setuid` bit set). The challenges in this exercise require students to bypass these weak access controls and recover the flag contained within each file.

Exercise 3: Protocol analysis. Students are given two key-agreement protocols in the standard "Alice and Bob" security protocol notation, Java source code of servers implementing these protocols, and raw network packet traces of clients communicating with these servers using these protocols. The protocols are vulnerable to a man-in-the-middle attack and a replay attack respectively. Students must discover and describe the attacks, implement them, and run them against the servers running on the VM; successful exploitation of each server reveals a flag.

Exercise 4: Web security. The VM runs a web server that hosts what at first sight appears to be an online furniture store; however, hidden inside this furniture store is a black-market website. Students must investigate this website, find the black-market interface and carry out a number of attacks. An SQL injection attack allows them to view all products stored in the backend MySQL database, including the black-market products and a flag. By analysing the cookies set by the furniture store, they can discover how the hidden website authenticates its users and gain access to it. The site displays a flag from a protected file that the students can submit to show that they accessed the hidden website. A file upload attack allows them to recover the MySQL server login details used by the black-market website; logging into the MySQL server using these credentials reveals another database containing a flag. A shell injection attack provides access to a shell running as the `www-data` user, revealing a final flag in a protected file in the file system. Non-flag questions in this exercise additionally cover XSS and CSRF attacks.

Exercise 5: Reverse-engineering. For this exercise, students must reverse-engineer four programs: two written in heavily-obfuscated Java and compiled into JDK bytecode, and two written in C and compiled into native code. All of these binaries behave as servers, and are listening on ports on the VM. Students must reverse-engineer the first three programs to recover a password expected to be entered into the program which, when entered into the copy of the program running as a server on the VM, causes the program to output a flag. The final program is an x86 binary for managing PGP keys, which also contains a backdoor; this binary listens on a port on the VM and runs as the `root` user. Students must find the backdoor by examining the binary in the free edition of the IDA disassembly tool [13] and use it to gain access to a root shell on the VM; a final flag is stored in a file in the `/root` directory.

3 A VM-Based Framework for CTF-Style Challenges

Our design goal was to create a VM-based framework suitable for use in a university cybersecurity course in which students could complete Jeopardy-style CTF challenges for course credit. While the use of VM-based exercises to support cybersecurity education is common (e.g., [3,24]), our aim was to design exercises that assess the full range of cybersecurity topics taught in the course in a fun and accessible way.

Our framework is based on a single VM that students download on their own hardware (e.g., their laptop) at the start of the course and import into the free and open-source VirtualBox virtualisation software [22]. This VM runs a Linux operating system containing several services, such as a web server, database server, and daemons running purpose-built insecure protocols, as well as many user accounts and complex, flawed access control configurations.

3.1 Flag Generation

When the VM boots for the first time, a *setup script* runs that generates the flags. These flags are written to particular locations (e.g., into the source code of a service which is then compiled, or into a password-protected MySQL database). The setup script then sets appropriate permissions on the generated files and binaries, and deletes any source code it compiled before deleting itself.

The setup script generates a random *VM identifier*, intended to be unique for each VM. To generate a flag for a particular question in an exercise, the function defined in Algorithm 1 is used: the VM identifier is concatenated with the *exercise number* and *question number*, and the resulting string is PKCS#7-padded and encrypted with AES using a 128-bit key (the *course key*) chosen by the course staff; the ciphertext can be represented as a string of 32 hexadecimal characters, and this is the flag that the student is required to submit. The validity of submitted flags is checked on the flag submission server using the inverse of the function in Algorithm 1: when decrypted with the course key, invalid flags will not be correctly PKCS#7-padded, or will be missing the `Ex` header.

```
Input: Course key $K$, exercise number $n_e$, question number $n_q$, VM identifier $v$
Output: Flag
function GENERATE_FLAG($K, n_e, n_q, v$)
    $flag\_data \leftarrow$ "Ex" $\|\ n_e\ \|\ n_q\ \|\ v$
    $plaintext \leftarrow$ PKCS7_PAD($flag\_data$)
    $ciphertext \leftarrow$ AES_ENCRYPT($K, plaintext$)

    ▷ Return ciphertext as string of 32 hexadecimal characters
    return BASE16($ciphertext$)
end function
```

Algorithm 1: Generating a VM-specific flag for a particular exercise and question

3.2 The Flag Submission Server

Students submit flags that are generated by the startup script by logging into the flag submission server (a website operated by the course staff) using their University IT account credentials and pasting the 32-character string into the input box for the appropriate exercise and question.

The flag submission server decrypts the ciphertext and verifies that it is a valid flag for the given question by checking the identifiers contained within the resulting plaintext. The server records details of the flag submission for the markers and instantly informs the student whether the submission was successful. Students are allowed an unlimited number of flag submission attempts for a particular question. To avoid the CTF-style challenges being perceived as competitions, students cannot check whether other students have submitted flags for a particular question.

An administrative page on the flag submission server shows the course staff details of all flag submissions for a given exercise and question, including the identity of the student submitting the flag, when the submission was made, and the identifier of the VM from which the flag originated. Students sometimes complete exercises using multiple copies of the VM (e.g., one on a laptop and one on a desktop computer), so we occasionally detect multiple VM identifiers for a particular student. However, students are explicitly instructed not to share their VMs with other students, so two students submitting flags with the same VM identifier is a possible indicator of plagiarism. The flag submission server does not reveal the results of this check to the student, but instead alerts the course staff of any irregularities on the administrative page; the course staff then follow this up with the implicated students individually (as we discuss in Section 7).

It would be possible for good students to download a fresh copy of the VM and reverse-engineer the flag generation program, giving them the ability to generate their own flags. However, we note that doing this would be harder than completing any of the exercises, in particular the existing reverse-engineering exercises. Additionally, we require students to submit descriptions of how they solved each exercise, which would be hard to do convincingly without solving the exercise as intended. A larger concern is students finding a way to gain root privileges on the VM and finding information that makes the exercises trivial; to counter this, we have carefully designed the exercises so that they rely on the secrecy of as few files as possible (e.g., in the web security exercise, the website's PHP code is not considered secret, and students are given access to it along with the exercise sheet).

4 Enhancing Teaching Through Stories

While gamification — in the form of point-scoring, team-based competition, freedom to fail and rapid feedback [16,26,23] — is popular in cybersecurity education, one must also consider the importance of storytelling and character development to ensure that the target audience remains engaged [25,15].

Building on our previous successes, we extended the VM framework described in Section 3 with a story for students to discover. The story is told through emails and news updates to a website in the VM, made possible with the integration of a local mail server and a custom-built *story engine* into the existing VM architecture; as the mail server and story engine are contained entirely within the VM, no support or maintenance is required from course staff, and each student controls their own version of the story. The story engine uses a set of XML files defining the content of emails and news updates to reveal to the student, and another XML file defining the conditions that must be met in order to progress along different paths of the storyline; it is therefore easy to change the story without modifying the internal mechanics of the VM. The story language features expressive logic permitting a range of complex scenarios (e.g., mutually-exclusive events, and different responses to different orders of actions taken by the student). If the VM has an Internet connection, we make it possible to provide telemetry for analytic purposes.

Fig. 1 depicts the interactions of the components of the VM framework described in Section 3 and the story framework described in this section and in Section 5. In brief, the process is as follows:

1. The setup script generates the flags and places them in files at the appropriate paths.
2. The setup script deletes itself.
3. The story engine sends the first emails, which can be read by the student from the player space via a mail client.
4. From the player space, the student interacts with the exercises and the web server, with the goal of obtaining the flags.
5. The flags are submitted to the flag submission server, an external website operated by the course staff.
6. The student emails the obtained flags to one of a number of characters, in order to progress the story.
7. The story engine periodically checks for new mail sent to the characters' inboxes on the mail server.

8. The story engine optionally sends the email to an external telemetry server.
9. Depending on which character received the email, the story engine advances the plot, updating the website hosted on the VM with appropriate news postings and sending new emails; the process repeats from step 3 onwards until the story ends.

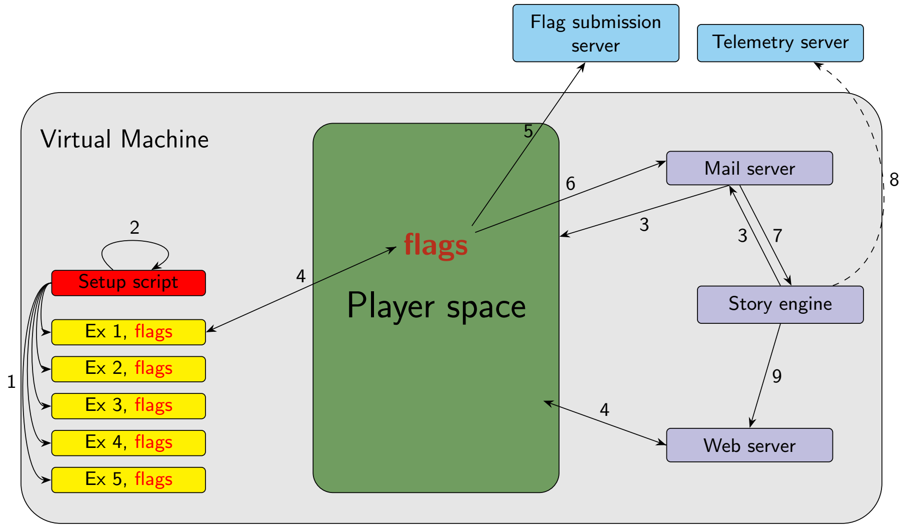


Fig. 1: The framework components: a self-contained virtual machine, a lightweight flag submission server and, optionally, a telemetry server

Students interact with the story engine by sending emails to the VM's mail server, which their mail client in the VM is preconfigured to use. The story engine runs as a `cron` job, periodically checking inboxes on the mail server, posting news updates to the website, and sending emails to the student from different characters in the story. Students advance the story themselves using flags they find by completing the course's regular exercises: after completing an exercise, the student must decide which character to email the recovered flags to. When the story engine receives an email, it will verify the enclosed flag and determine which character's inbox the email was sent to. Based on the choices made by the student, the engine progresses the story by posting further news updates on the website and sending new emails to the student. This feedback loop gives students the ability to control what happens in the story using an approach for increasing player engagement that is commonly found in large-scale computer games.

Our story follows the classic Hollywood story arc structure: a trigger event, crossing a threshold, overcoming obstacles, a setback, and then a final push.

Each of these five stages of the story corresponds to a two-week exercise. The students start out as a new security employee at a company, where they are asked by their line manager to complete some decryption code and find some flags, and are warned to keep the flags secret. Meanwhile, the students receive a second email from a mysterious stranger telling them that there is something wrong at the company, and that they should instead send the flags to them. The story engine ensures that the students can only choose one of the two options. As the story progresses, they discover that the company is being used as a front for a black-market website (the Cotton Highway) and, after the police become involved, they must decide what to risk and who to side with.

We introduced the story as a trial extension to the course VM, giving the students the option of following the story or not. The concept of the story and how students could interact with it was described in lectures, and a statistically-insignificant bonus mark was offered for following the story. To evaluate the value of the story, we performed surveys for students who followed it and those who did not. Analysis of the results showed that students who followed the story felt engaged, while those who did not felt that the concept was good, with positive feedback. We also evaluated the difference in obtained marks between the two groups of students, and found that students who followed the story achieved significantly higher marks on average than those who did not. To rule out the possibility that stronger students chose to follow the story and weaker students chose not to, we compared the marks each student achieved on this course with the marks they achieved on other courses. We found that students who followed the story did much better than their marks from other courses would predict, whereas the marks of students who did not follow the story were consistent with their marks in other courses. This suggests that following the story did improve student performance. A more detailed analysis is presented in Section 7.2.

Further extensions have been made to the story engine and the way the course is gamified. For example, one extension to the story engine implements *welfare flags*: should a student have mitigating circumstances that prevent them from completing a particular exercise, a special flag can be given to them that progresses the story beyond that exercise; this ensures that students are never put at a disadvantage to their peers. Other extensions to the story engine make it possible to define multiple paths forward for a given set of conditions (providing some randomness to the consequences a student faces for sending flags to a particular character), and artificially delay emails being sent and news stories being posted on the website (simulating a human being on the receiving end of the student's emails, and giving the story an organic nature). This allows self-paced learning, and a more bespoke student experience regardless of whether the student took similar choices to their peers. We also improved the VM's reporting ability such that if the VM was unable to send telemetry data due to a lack of an Internet connection, it would retry when connectivity became available.

5 The Story Engine

In order to convey a story to the students, we required an immersive way of communicating with them. We therefore made the story dynamic, reacting to the actions taken by the student so their experience would feel personalised. To achieve this, we developed a Java-based story engine that would not only monitor the decisions made by the student, but also tailor its output based on these decisions. To further enhance the immersion, we added a company website (hosted on a web server running locally on the VM), featuring a news section where the story engine would add new updates as the story progressed. These updates could be relevant to the decisions taken, or presented in a more general format.

One aim of this extension to the VM was to make the story easy to write and maintain while requiring no particular programming knowledge. This is achieved through the use of an XML-based configuration file for the story engine, which we refer to as a *story map*. The story map encodes the different paths that can be taken through the story and the objectives that have to be completed in order to progress along each path. State attributes in the story map are used to track progress through the story.

In the event that a student abandons the story before it ends, they can to return to where they left off at any time; alternatively, they may use a welfare flag (an extension to the story engine described in Section 4) given to them by the course staff to advance the story to a specific point.

5.1 Story Map

The story map is an XML-encoded file; an overview of its structure is shown in Fig. 2.

The `<exercise>` element provides a logical separation between the different parts of the story; as our course was already written around a series of five exercises, we chose to break our story into five parts, each corresponding to a particular exercise, although this is customisable. Inside each `<exercise>` element are several `<event>` elements defining events that occur within the story, split up into smaller `<tasks>` and the `<required>` conditions that trigger them.

The `<tasks>` element contains a list of tasks that the story engine performs when an event is triggered. These are either `<email>` tasks (instructing the engine to send an email to the student), or `<news>` tasks (instructing the engine to update the website with a new news story). Both task types point to auxiliary XML files containing the text of the emails to be sent and news stories to be posted.

The `<required>` element contains a list of conditions that must be met before the event's `<tasks>` are executed. These can either be `<token>` elements (indicating that a specific flag must have been sent to a specific email address, defined in the `sent_to` attribute), or `<finished>` elements (indicating that a specific event must have already occurred). Nestable `<AND>`, `<OR>` and `<NOT>` elements allow conditions to be combined or negated, allowing for arbitrarily

```
<story_map>
  <exercise>
    <event>
      <required>
        <!-- Arbitrarily complex conditions, e.g.: -->
        <AND>
          <finished ... />
          <OR>
            <token ... />
            <token ... />
          </OR>
        </AND>
      </required>
      <tasks>
        <!-- Perform these actions when conditions are met -->
        <email ... />
        <news ... />
      </tasks>
    </event>
  </exercise>
  <news_stories>
    <!-- Static news story definitions -->
    <static_story ... />
  </news_stories>
  <extras>
    <!-- Character definitions -->
    <users ... />
  </extras>
</story_map>
```

Fig. 2: Overview of the story map file structure; some element types (e.g., `<users>`, `<news>`, `<static_story>`) additionally rely on external XML files

complex requirements for triggering tasks. The story engine can also randomly select an event to trigger if multiple events are triggered by the same conditions; this gives each student a more bespoke experience and allows the otherwise-linear story to appear to progress organically.

The `<event>`, `<email>`, `<news>` and `<token>` elements contain a `complete` attribute; these are used to track which events and tasks have already been completed and which flags have already been sent to the story engine, thus tracking the current state of the story. Their values are initially set to `false`, then updated to `true` as appropriate by the story engine as the story progresses.

The `<users>` element points to another auxiliary XML file that defines characters by their account name and password on the VM's mail server; the story engine periodically checks the inboxes of these characters and reacts to emails received from the student, as described below. The `<news_stories>` element defines *static stories* to be posted on the news page at particular times, and is described in more detail in Section 5.4.

5.2 Receiving Flags

The students progress the story by emailing the flags they find when solving exercises to one of the story's characters via a mail server which runs locally on the VM. The story engine polls the inbox of each character defined in the story map; received emails are searched for candidate flags (32-character strings of hexadecimal characters, matching the flag format described in Section 3.1). Candidate flags are decrypted using the course AES key to check their validity. When an email has been processed, and telemetry data logged, the email is deleted from the inbox; if there is no Internet connectivity, the email is deleted only after a copy has been sent to the telemetry server. When a valid flag is found, the story engine extracts which exercise and question number the flag corresponds to and identifies any requirements involving this flag in the story map. If the flag has been sent to the correct character (as defined in the requirement's `sent_to` attribute), the requirement's `complete` attribute is set to `true`.

5.3 Sending Emails

The most common task is for the story engine to send an email from one of the characters to the student via the VM's mail server. Separate XML files encode the body of each email to be sent, along with its subject, sender and recipient (for team exercises, where each student is a different character). The story engine completes an `<email>` task by reading the XML file defined by its `path` attribute and sending an email based on its contents. By taking advantage of the logical operators available in the `<required>` element, different emails can be sent depending on the choices made by the student; a common pattern is "send email w if flag x was received by user y, but not if event z has been completed".

5.4 Posting News Stories

The website used in one of the final exercises features a news page containing a client-side script that populates the page with all news updates encoded as XML files located within a specific public directory on the web server. The story engine can post updates to this page as the story progresses (*dynamic stories*) simply by copying the XML file defined by the `<news>` element's `path` attribute into the public directory; when the student reloads the page, the update becomes visible.

It is also possible to define *static stories* unrelated to the overall story in the story map via `<static_story>` elements, which are posted to the website's news page at predefined times regardless of how the student is engaging with the story. Static stories give a sense of realism to the website, and provide a regular supply of news updates, giving the student a reason to keep checking the news page.

5.5 Telemetry

To allow the course staff to observe the decisions taken by the students, the story engine sends a copy of any emails sent to the story's characters, along with the student's VM identifier (as described in Section 3.1), to an external telemetry server controlled by the course staff.

Given that the student must include flags they find in the emails they send in order to advance the story, it is possible to combine these emails with the telemetry functionality to track which flags have been discovered by the student, rather than requiring the students to submit their flags to a separate flag submission server operated by the course staff. This provides a way of forcing students to interact with the story in order to gain credit for completing exercises; however, this was undesirable for our iteration of the course, which featured the story as a trial extension.

6 The Story

Progression and storytelling, as identified by Stott and Neustaedter [26], are two key concepts of game design that can be successfully applied to teaching and learning. This approach, however, is less prevalent in cybersecurity education. Our objective was to create an exciting, alluring and believable story, with the eventual aim of increasing student engagement with the course.

Our story follows the three-act screenwriting structure [27] commonly used in Hollywood scripts. Each exercise corresponds to a separate stage within the story arc: a call to adventure, crossing a threshold, overcoming obstacles, a setback, and a final push. This ensures that the story arc keeps the students engaged and excited to progress through the story.

One consideration when developing the plot of a story is whether the nature of the events is appropriate for the target audience. We applied the BBFC [2] and

ESRB [10] classification schemes to our story, and found that its mild references to drugs and violence and moderate threat would likely cause it to be rated "12" or "Teen" respectively, making it appropriate for undergraduate students. The story engine is agnostic of the story map and its auxiliary files that define the emails and news updates shown to students, so our framework also supports plots that suit alternative audiences, for example younger teenagers or children.

6.1 Setting and Characters

The story takes place at Sensible Furniture, a fictional furniture company with a dark secret that is revealed as the story progresses. The plot features a protagonist — played by the student — and five other characters whose objective is to manipulate the protagonist into carrying out actions that follow their agendas:

Employee 427. Sensible Furniture's new cybersecurity advisor, and the main character and protagonist of the story, played by the student. In defining the protagonist, we chose a number instead of a name in order to (a) ensure all students can identify with the character, and (b) introduce a sense of impersonal coldness to the setting in which the story takes place.

Jak Kinkade. The CEO of Sensible Furniture. They introduce Employee 427 to their new working environment, reappearing towards the end of the story to give Employee 427 a final push towards a dangerous path.

Nik Adler. Employee 427's line manager at Sensible Furniture. Their main role is giving Employee 427 their daily tasks, and keeping them on track.

Charle Garcia, aka Chimp. An employee of Sensible Furniture. At the start of the story, Charle is known by the alias "Chimp" and approaches Employee 427 to convince them to join their side, hinting that the company's bosses are suspicious characters. Chimp's character represents the outcast, the undercover potential ally who will guide Employee 427 along the path of righteousness.

PC Thomson Gazal. A police officer, appearing when Employee 427 experiences a crisis and the available options seem to be limited. Thomson presents a new opportunity for Employee 427 to choose the path of righteousness.

Carol Miller. The IT administrator of Sensible Furniture by day, and a questionable character by night. She appears towards the end of the story, in order to give it a new twist. Carol offers Employee 427 a different perspective on the events that have unfolded and a new opportunity: a role in the criminal underworld.

6.2 Plot

An overview of our story's structure, and how the different stages of the story map to the course's exercises described in Section 2.2, can be seen in Fig. 3. It depicts all of the story's characters, and all information — flags, emails and news updates — that can be exchanged between the student and the story engine.

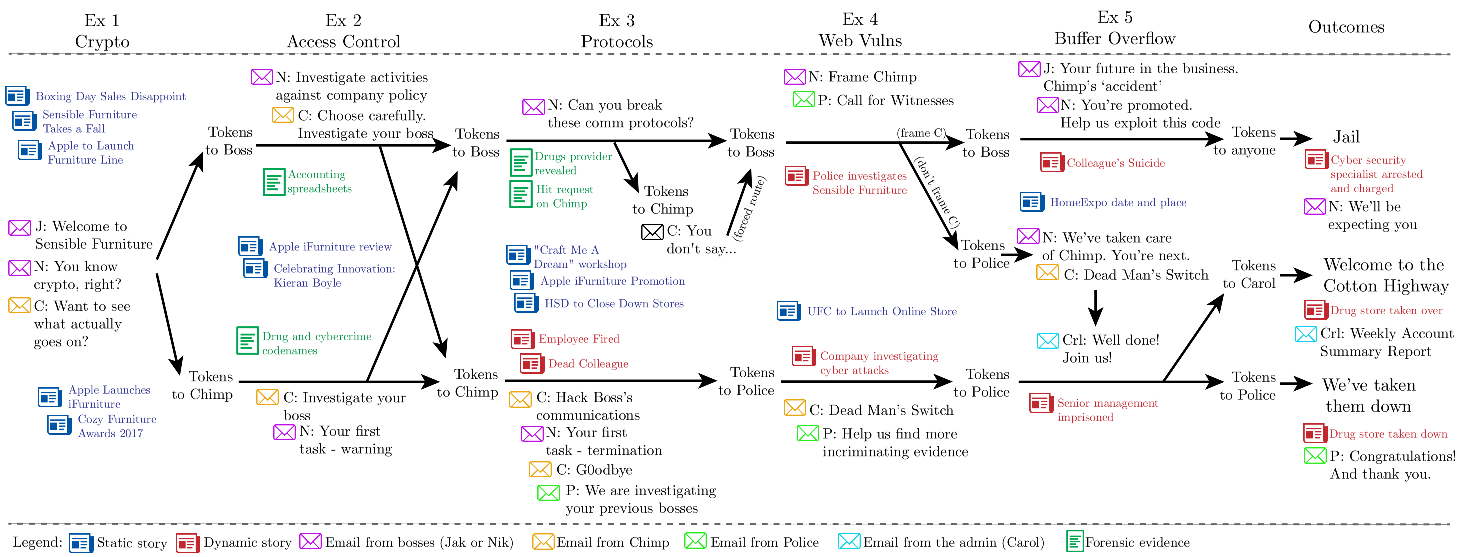

Ex 1
Crypto
Ex 2
Access Control
Ex 3
Protocols
Ex 4
Web Vulns
Ex 5
Buffer Overflow
Outcomes
Boxing Day Sales Disappoint
Sensible Furniture Takes a Fall
Apple to Launch Furniture Line
J: Welcome to Sensible Furniture
N: You know crypto, right?
C: Want to see what actually goes on?
Apple Launches iFurniture
Cozy Furniture Awards 2017
Tokens to Boss
Tokens to Chimp
N: Investigate activities against company policy
C: Choose carefully. Investigate your boss
Accounting spreadsheets
Apple iFurniture review
Celebrating Innovation: Kieran Boyle
Drug and cybercrime codenames
C: Investigate your boss
N: Your first task - warning
Tokens to Boss
Tokens to Chimp
N: Can you break these comm protocols?
Drugs provider revealed
Hit request on Chimp
Tokens to Chimp
C: You don't say...
(forced route)
"Craft Me A Dream" workshop
Apple iFurniture Promotion
HSD to Close Down Stores
Employee Fired
Dead Colleague
C: Hack Boss's communications
N: Your first task - termination
C: G0odbye
P: We are investigating your previous bosses
Tokens to Boss
Tokens to Police
N: Frame Chimp
P: Call for Witnesses
(frame C)
(don't frame C)
Police investigates Sensible Furniture
UFC to Launch Online Store
Company investigating cyber attacks
C: Dead Man's Switch
P: Help us find more incriminating evidence
Tokens to Boss
Tokens to Police
Tokens to Police
J: Your future in the business. Chimp's 'accident'
N: You're promoted. Help us exploit this code
Colleague's Suicide
HomeExpo date and place
N: We've taken care of Chimp. You're next.
C: Dead Man's Switch
Crl: Well done! Join us!
Senior management imprisoned
Tokens to anyone
Tokens to Carol
Tokens to Police
Jail
Cyber security specialist arrested and charged
N: We'll be expecting you
Welcome to the Cotton Highway
Drug store taken over
Crl: Weekly Account Summary Report
We've taken them down
Drug store taken down
P: Congratulations! And thank you.
Legend:
Static story
Dynamic story
Email from bosses (Jak or Nik)
Email from Chimp
Email from Police
Email from the admin (Carol)
Forensic evidence

Fig. 3: An overview of our story, and how it fits into the exercise structure of our course

The arrows show the possible paths the student can follow at any given point, together with their effects on the story's progression.

The first exercise requires the student to demonstrate an understanding of cryptography by writing code in the VM that decrypts certain encrypted files containing flags. This coincides with the opening stage of the story's plot, in which Employee 427 is welcomed to the company by CEO Jak Kinkade and introduced to Nik Adler, their line manager. Nik instructs Employee 427 to send the decrypted files to him. Moments later, Employee 427 receives another email from a person named Chimp (see Appendix A.1), suggesting that the management should be considered suspicious, and are "in cahoots with the underworld". Chimp requests that Employee 427 sends the decrypted files to them instead, and offers to join forces with Employee 427 to take down the management. This provides the student with the choice of following one of two paths: either be a good employee by sending the files to Nik, or risk trusting Chimp instead.

The second exercise requires the student to demonstrate an understanding of access control on Linux file systems and mount attacks against common access control misconfigurations to recover flags from restricted-access files on the VM. This exercise coincides with the second stage of the plot, in which Employee 427 must learn more about the company's other employees by inspecting the contents of their home directories. The home directories contain clues that there is illegal activity occurring within the company, but give no indication of who is involved; this is intended to make the student question their previous decision. They find accounting spreadsheets in Nik's home directory with some suspicious entries: code names for drugs and cybercrime. In Chimp's home directory, some files containing information about drugs and cybercrime transactions can be found. At this point in the story, the student can change paths by emailing their flags to the character they chose not to email in the first stage of the story.

The third exercise involves breaking some encrypted communication protocols and requires the student to launch attacks against implementations of those protocols running on the VM in order to recover flags. This coincides with the third stage of the plot, which begins to seal Employee 427's fate. Employee 427 intercepts and decrypts secret messages being sent over the company's network; the messages reveal who the drugs provider is and, disconcertingly, a request for a hit on Chimp. At this stage of the story, if Employee 427 is cooperating with Nik, they cannot switch sides: if they try to email the recovered flags to Chimp, Chimp will bitterly refuse to accept them and tells Employee 427 to continue on their current path. If Employee 427 is collaborating with Chimp, a setback appears: Employee 427 is fired from Sensible Furniture, and receives an email from Chimp informing them that their activity has been discovered and directing Employee 427 to communicate with PC Gazal. The company website's news page will then display two new updates: one announcing that Employee 427 has been fired, and another reporting that Charle Garcia, a former employee, has "committed suicide".

The fourth exercise focuses on web security, and requires the student to perform a variety of attacks against an interactive website hosted on a web server running on the VM, leading to the discovery of the Cotton Highway, a black-market website hidden behind it. This coincides with the fourth stage of the plot, in which the plot thickens on both paths. On the path where Employee 427 cooperates with Nik, they receive a surprising request: investigate and attack the company website, and, if illegal activity is found, frame Chimp (who is still alive on this path). At the same time, PC Gazal introduces himself, asking Employee 427 to become a confidential informant. Employee 427's situation seems very grim on this path, and Gazal's introduction heralds a chance at redemption: the student can choose to continue down what seems like a dark road and frame Chimp for the illegal activity on the website, or turn to the police. On the path where Employee 427 collaborated with Chimp before their premature death, Employee 427 receives a "dead man's switch" email set up by Chimp to be sent automatically in the event that something bad were to happen to them. The email reveals Chimp's real identity, and contains information incriminating Nik and Jak in drug trafficking and hiring hitmen. At this point in the story, only two courses of action are apparent for Employee 427: either continue working with Nik and Jak and fall in with the criminal underworld, or cooperate with PC Gazal and hope to be exonerated.

The fifth exercise requires the student to reverse-engineer a number of programs, including a service that runs as the `root` user on the VM, and exploit a vulnerability to gain superuser access to the VM. This coincides with the final stage of the plot, which places the spotlight on the heretofore background character Carol, Sensible Furniture's IT administrator. Carol reveals that she operates the Cotton Highway, the company's hidden black-market website. If the student chose to collaborate with PC Gazal in the previous stage, she offers Employee 427 a potential third course of action: use their newfound superuser access to the Sensible Furniture systems to join the criminal underworld and supplant Nik and Jak as the drug kingpin. With the police closing in on Nik and Jak, Carol's offer is the last twist of the story. The news of Charle's apparent suicide appears on the company website. However, it is later revealed that this was a contract killing arranged by Nik and Jak.

The story's ending depends on the decisions made by the student:

- If Employee 427 remains loyal to Nik and Jak, they have no escape and, regardless of who they choose to turn to, they are ultimately sent to jail along with Nik and Jak. Nik menacingly informs Employee 427 that they are expecting his arrival (see Appendix A.2).
- If Employee 427 sides with Chimp and turns over the last set of flags to PC Gazal, a happier ending occurs: Nik and Jak are imprisoned, and the Cotton Highway is shut down. PC Gazal also reveals that Chimp was his fiancée, thanking Employee 427 for their help in apprehending Chimp's murderers.
- If Employee 427 accepts Carol's offer, they take over the Cotton Highway, supplanting Nik and Jak as the drug kingpin. If Employee 427 previously betrayed the managers, they receive a threatening email from Nik, letting them know Chimp had been "taken care of", and that Employee 427 is the next target on the list.

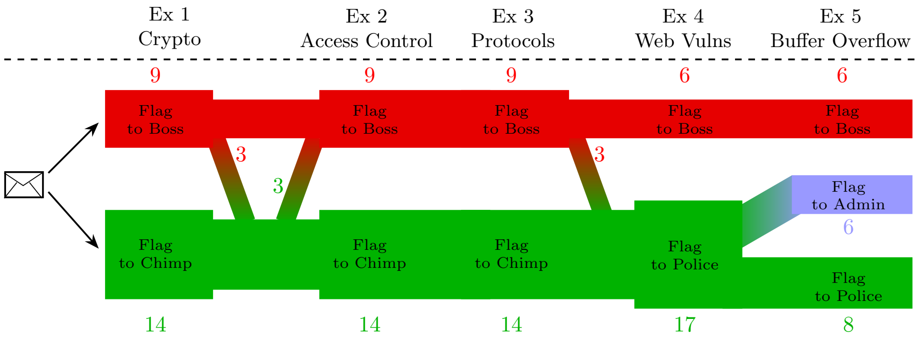


Fig. 4: The decisions taken by students on the course to progress through the story

6.3 Student Choices

Fig. 4 depicts the decisions made by the 23 students who chose to engage with the story, as reported to our telemetry server. The paths are colour-coded as follows: red paths indicate cooperation with Nik, green paths indicate collaboration with Chimp and PC Gazal, and blue paths indicate acceptance of Carol's offer.

At the start of the story, 9 students chose to follow Nik's orders, and 14 chose to trust Chimp. However, in the second stage of the plot, where they are given some clues that illegal activity is occurring within the company, 3 students from each path chose to switch sides. This behaviour substantiated our expectation that students would not blindly follow a specific path to the story's conclusion, but would instead doubt their choices in response to events that occurred within the story, or would simply switch sides just to see what (if any) effect their actions would have on the plot.

Allegiances remain steady for the third stage of the plot, while the fourth stage sees 3 more students shifting from following management orders to refusing to frame Chimp for the illegal behaviour occurring within the company and instead cooperating with the police. At the end of this stage, we see only 6 students remaining loyal to the management, while 17 of them are cooperating with PC Gazal in order to find incriminating evidence on Nik and Jak.

The appearance of Carol in the final stage of the plot divides the students who had previously been cooperating with the police, with 8 choosing to continue cooperating, and 6 choosing instead to become a drug kingpin. We also note that 3 students abandoned the story between these stages. The roughly-even split of story endings emphasises that students felt that all three paths were viable, validating the quality of writing and our story design.

7 Evaluation

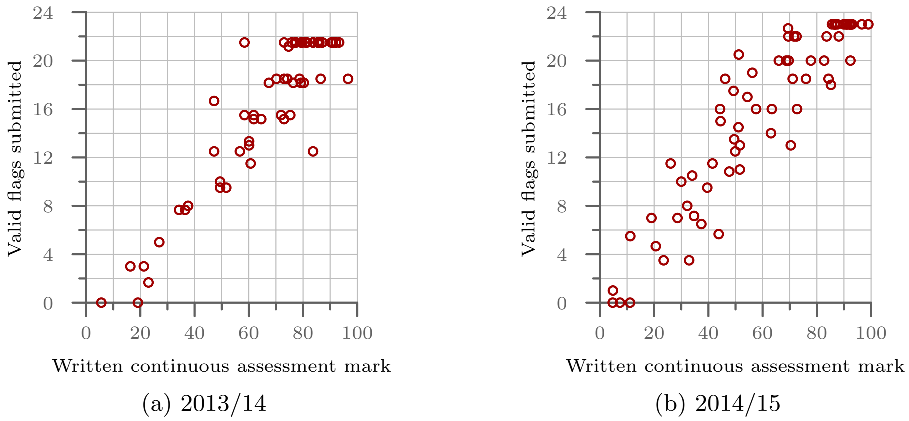
We now perform an analysis of the effect of introducing the VM and CTF-style challenges to our cybersecurity course, and then assess the value introduced through the addition of a story to this VM. Unlike the introduction of the story, the introduction of the VM was a major change to the course: all of the continuous assessment required students to interact with the VM and submit valid flags embedded within it. We chose to make interacting with the story optional to avoid any unexpected negative impact that such an experimental component could have on the course's large cohort if it were made compulsory.

From a cohort of 144 students, a self-selected sample of 23 chose to follow the story; while this number was smaller than we had hoped, it provided enough data to carry out an interesting analysis. The low uptake could be explained by a lack of regular promotion of the story, or by students being less inclined to involve themselves in an aspect of the course that was not mandatory. Students who chose to follow the story were not given an inherent advantage in the continuous assessment compared with those who chose not to: as discussed in Section 4, we awarded a statistically-insignificant bonus mark for the continuous assessment component of the course as an incentive to interact with the story, and none of the paths in the storyline contained hints about how to proceed with the exercises.

7.1 Assessing the Impact of Introducing a CTF-Style VM

We first consider correlations between flag submissions and continuous assessment marks following the introduction of the VM in the 2013/14 iteration of the course. Recall from Section 2.1 that to complete an exercise, students were required to submit not only flags they recovered from the VM, but also written descriptions of vulnerabilities that were exploited to recover the flags and proposals for fixing those vulnerabilities. We leveraged the data submitted to the central flag submission server to compare how successful a student was at recovering flags (which was assessed automatically by the flag submission server) with the quality of their written answers (which was assessed manually by the course staff). From this, we can consider whether it is possible to replace manual marking of exercises with a fully-automated solution based solely on flag submissions.

Fig. 5 shows a set of scatter plots of each student's continuous assessment mark compared with the number of valid flags they submitted, across the 2013/14 and 2014/15 iterations of the course. Each point reflects a single student; the number of flags submitted (on the y axis) is weighted for difficulty. We observe a strong correlation between the number of flags submitted and the continuous assessment mark, as shown in Table 1. We note that this correlation is not uniform across all grades: it is weaker for students who achieved the highest grade (awarded for a mark of 70 or greater) than it is for those who achieved lower grades.



(a) 2013/14

(b) 2014/15

Fig. 5: Scatter plots of students' written continuous assessment marks and the number of valid flags they submitted

	Iteration of course	
	2013/14	2014/15
All marks	0.92	0.93
Marks ≥ 70	0.37	0.68
Marks < 70	0.91	0.90

Table 1: The correlation between the number of valid flags submitted by a student and their written continuous assessment mark for the course

We found that students who submitted a valid flag for a given question also received at least a good mark for their written answer for that question. The difference between good and outstanding marks can be attributed to higher-attaining students demonstrating a deeper understanding of the course content and cybersecurity issues associated with a particular question. For instance, when students were asked to fix flaws in the protocols in Exercise 3, only students with a strong understanding of the protocols could propose and describe adequate solutions, regardless of whether they were able to recover the corresponding flags.

We note that the 2014/15 iteration of the course was simpler than other iterations: the reverse-engineering exercise was unassessed, as were the written questions in the protocol analysis exercise requiring students to propose fixes for the protocol vulnerabilities. For this iteration, we observe a stronger correlation between the continuous assessment marks and the number of valid flags submitted. As with 2013/14, we observe a weaker correlation for students who achieved the highest grade, although it is not as pronounced.

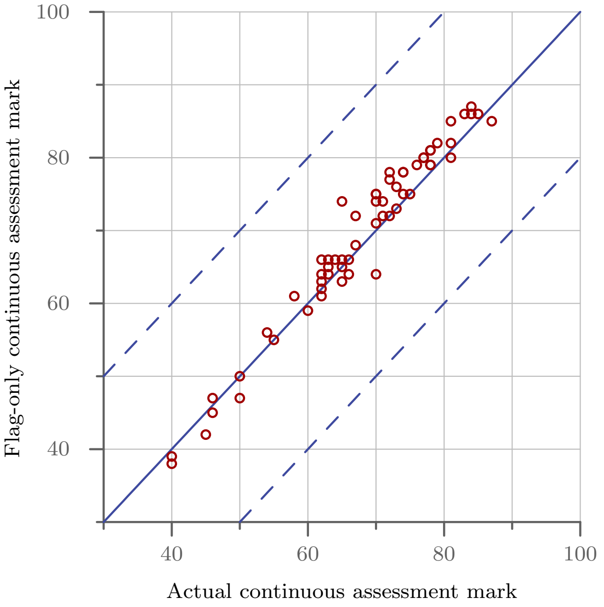


Fig. 6: Comparison of 2013/14 students' actual continuous assessment marks and the marks they would have been awarded had they only been assessed on their valid flag submissions

Fig. 6 shows the continuous assessment marks that students achieved in the 2013/14 iteration of the course versus the marks they would have attained if the course were assessed purely based on flag submissions. The mean difference between marks is 1.5, and the standard deviation is 2.5. Given that grade boundaries occur at 10-mark intervals between 40 and 70, most students would have received the same overall grade, although there are some exceptions (particularly at the highest grade boundary, where the maximum difference was 9 marks).

From a student attainment perspective, our analysis shows a high correlation between a student's ability to complete CTF-style challenges and their continuous assessment marks, suggesting that flag acquisition is a useful assessment technique for academic cybersecurity courses. However, our data also shows that assessing a student's ability to complete such challenges in isolation is not an appropriate way to assess the level of understanding of cybersecurity topics.

One facet of teaching not addressed by this analysis is the pedagogical value to students of formulating detailed, long-form answers to questions and receiving feedback on the assessments they complete. This requires a degree of self-reflection on behalf of the student. Higgins et al. [14] provide evidence that students are driven by more than just the potential of attaining high marks, benefiting from feedback that allows them to further develop their engagement with the subject. In our course we provide detailed, personalised feedback for all written answers, something that students indicated was highly valued and a reason for their overall satisfaction with the course; consequently, our course was ranked third-best for feedback from assessments amongst all courses offered by the School. Using flag-only assessment would both deprive students of this feedback and decrease their satisfaction with the course. However, we must also acknowledge that manually marking submissions and providing this rich feedback is highly labour-intensive on the part of the course staff, and its viability depends heavily on the course's staff-to-student ratio.

In any taught course, there is the potential for students to plagiarise, whether accidentally or intentionally; the same potential exists when utilising flag-based assessment, although we can detect many incidents of such plagiarism automatically thanks to the unique VM identifier built in to the flag structure (see Section 3.1). Across all iterations of the course, we encountered three cases where groups of students had submitted flags with the same VM identifier, or the VM identifier was inconsistent with the student's previous flag submissions:

- In the first case, a group of four students submitted the same flag and similar written answers for a particular exercise. When interviewed, they admitted to copying each other's work; each student was given a warning and mark of 0 for the entire exercise.
- In the second case, two students submitted the same flag, but their written answers differed. After review, it transpired that the students were using low-powered devices with a small amount of internal storage, and were sharing the same instance of the VM on an external disk to save space; both students were given advice on how to avoid any potential repercussions.
- In the final case, a student had posted a screenshot of their VM's current state on a social media website showing that they had completed a particularly challenging question, in an apparent attempt to impress their peers;

> the screenshot included the flag they had managed to recover by completing the question, and some enterprising students had transcribed the flag from the screenshot and submitted it to the flag submission server ahead of the student who had recovered the flag. The students concerned were penalised for their lack of foresight or ethics as appropriate.

Conversely, we also occasionally encountered cases where students submitted different flags but identical written answers. We note that had we employed flag-only assessment, we would have been unable to detect this sort of plagiarism, and would thus have created a favourable environment for plagiarism in which students could feign understanding of the cybersecurity issues underpinning an exercise simply by following instructions for recovering flags from their own instance of the VM from a more knowledgeable student. Overall, we saw much lower rates of plagiarism on our course than on other courses that had a continuous assessment component contributing 20% to the final course mark; when asked informally, some students commented that the requirement to submit written answers alongside flag submissions meant that they saw no benefit to plagiarising flags.

7.2 Assessing the Impact of Introducing a Story

We now consider how the course was impacted in the 2016/17 academic year by the addition of a story that was optional for students to follow.

We first compare the final course marks awarded to students who did and did not follow the story. Table 2 shows a summary of the final course marks (out of 100) for the 2016/17 iteration of the course; a histogram of these marks is shown in Fig. 7. Fig. 7 shows that the minimum mark for students who interacted with the story was 50, with the majority achieving a mark of 70–80. We also note that of the 20 students with the highest continuous assessment marks, 15 had interacted with the story rather than completing the exercises solely through flag submission and written answers.

Student category	Population	Average mark
Followed story	23	72.35
Did not follow story	121	61.27
All students	144	63.04

Table 2: Summary of final marks for the 2016/17 iteration of the course

To assess the impact of the VM's story on student engagement with the 2016/17 iteration of the course, we released an additional assignment during the University's "reading week" — when no teaching takes place, and students are

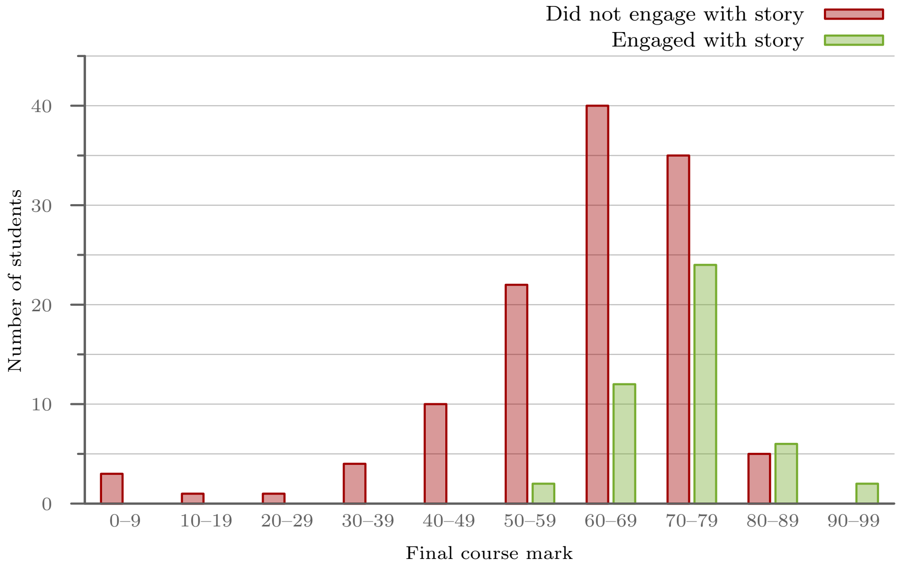


Fig. 7: Histogram of final marks for the 2016/17 iteration of the course

expected to engage in independent learning — to research and write a report on current cybersecurity issues. We cross-referenced the report each student wrote with data sent by their VM to the telemetry server, and found that reports written by students who engaged with the VM's story contained on average 74% more words than reports written by those who did not. We note that marks allocated to this assignment were given simply for completing the report; we believe that writing a longer report is a good indicator of greater engagement with the course, and therefore that interacting with the story is correlated with increased engagement.

From the telemetry data, we discovered that students generally submitted their flags to the flag submission server before using them to progress the story via the story engine in the VM. 7 students immersed themselves in the story more than others, and engaged in conversation with the story engine; a sample of the emails they sent can be found in Appendix B. This suggests that the story was believable and engaged students, encouraging them to spend more time on course-related activities.

Finally, to account for individual student ability, we analysed the difference between each student's final mark in our course and their average mark across all courses for the previous year. We removed from the analysis 7 students who did not have an average mark for the previous year (this usually occurs in welfare cases), and 2 students who did not submit any continuous assessment or sit the exam for our course. Following this, we tested for any differences between the sets of students who did and did not interact with the story. The results are

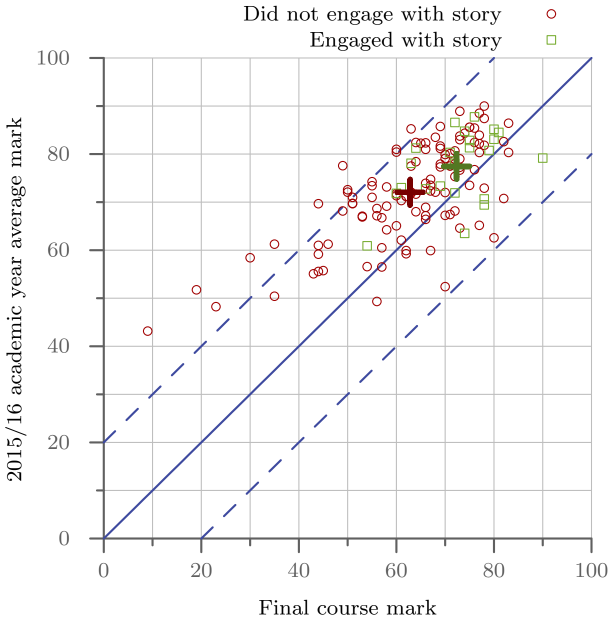


Fig. 8: Scatter plot of students' final marks for the 2016/17 iteration of the course versus their average marks for the previous year

shown in the scatter plot in Fig. 8; the analysis showed that the mean difference for students who did not interact with the story was -9.22 (i.e., their final mark for our course was on average 9.22 marks lower than their previous-year average mark), whereas the mean was only -5.09 for those who did interact with the story (p-value < 0.05). This shows that interaction with the story did in fact improve student performance, and the difference in means is not likely a result of chance. The lower average mark for both sets of students could be explained by the natural increase in difficulty of courses in successive years of a degree programme.

To gauge how well the story was received by the cohort, we requested the students' participation in an online post-course survey about the story; we received 52 responses. We received excellent feedback: those who interacted with the story gave an average score of 5.5/10 when asked how much the story increased their engagement with the course (with 6% stating that it significantly increased their level of engagement), and gave an average score of 7/10 when asked how fun they found the story (with 12% rating it as extremely fun).

In the same survey, we asked students who began the story whether they saw it to completion; 38% did. The majority of students completed the story in 1 hour, and it was, predominantly, the curiosity about how the story would progress that motivated them. Those who did not engage with the story at all gave different reasons for their choice, e.g. more important priorities in other courses. Amongst those who did not engage with the story, there was a variation in the expected time required to compete the story, with most suggesting 2 hours to more than 3 hours. This disparity — between the perception of how time-consuming it would be to interact with the story amongst those who chose not to, and how time-consuming it actually was amongst those who did — may have influenced take-up; had we informed the students in advance of the expected amount of time it would take to complete the story, it is possible that more students would have participated. Some students were unable to complete the story regardless of their desire to do so, as they had not recovered a sufficient number of flags to advance the plot for at least one of the exercises, and did not have extenuating circumstances requiring us to issue a welfare flag allowing them to advance the plot forcefully. These issues could both be addressed in future iterations of the course.

Overall, the consensus of the concept of the story was very positive, with 97% of students who engaged with it and 84% of those who did not agreeing that the story was a good idea; students commented that it did (or would) make the course more interesting and increase engagement, with an alternative reward to what students would typically expect. When asked if it would be a good idea to add a story framework to other courses taught in the School, 63% of students who engaged with the story and 59% of those who did not agreed, suggesting a list of courses that could benefit from the introduction of a story. This suggests that not only did our story framework add value to our course, it could potentially add value to others too.

Finally, when asked about the story itself and how satisfied they were with its conclusion, the students gave an average rating of 3.3/5, with 17% being extremely satisfied. An average score of 5.9/10 was given for the plausibility of the plot, with 3% considering it extremely believable. When asked how interactive and captivating they found the story, scores of 6.2/10 and 6.3/10 were given respectively, with 13% of students stating that they found the story extremely captivating and were keen to see how the story progressed. When asked to rate the quality of the storytelling, an average score of 7.5/10 was given, with 18% rating the story as very well-written. These factors may have contributed to the level of engagement with the story and the course in general. A sample of the comments we received is given in Appendix C.

From the results of the survey, we conclude that students who engaged with the story were encouraged through their captivation with and curiosity about the story to devote more time to the course and therefore complete it with higher continuous assessment marks. Even those who did not engage with the story considered it a worthwhile addition to the course, so we suspect that, if the expected participation requirements had been made clearer from the outset, we would have observed greater levels of engagement. The survey demonstrates the positive impression of the story on the 2016/17 cohort and sheds light on why only a relatively small number of students completed the story; the reasons for this can easily be addressed, and the survey gives us confidence that interacting with the story could be made compulsory in the next iteration of the course.

8 Conclusion

In this article, we have presented a virtualised framework for integrating Jeopardy-style CTF challenges into cybersecurity courses, along with a narrative extension to this framework. Students run the framework on their own hardware, eliminating the need for course staff to maintain a complex backend infrastructure. The narrative extension is highly parameterisable, making it easy to modify or rewrite the story between iterations of a course. We have evaluated both the framework and the narrative extension by integrating them into the continuous assessment component of our own 11-week introductory cybersecurity course, the former in the 2013/14 academic year and the latter in the 2016/17 academic year; this allowed us to evaluate the impact of the framework and chosen story on students' academic performance and the broader student learning experience.

Through an analysis of the continuous assessment and final course marks obtained by the students over two iterations of our course, we found that a student's ability to recover flags in the CTF challenges correlated closely with their overall performance on the course; as our framework generates and embeds flags and validates flag submissions automatically, this allows for a degree of automated marking to be employed, although we would not advocate this as the sole method of assessing students' level of understanding of cybersecurity topics. Through an analysis of students' marks for both our course and other courses, we found that students who engaged with the story in our course performed better than their average mark in other courses would have predicted, whereas those who did not engage with the story performed no better (or worse); post-course surveys indicated that students valued the plausibility and entertainment value of the plot, thus providing evidence that the introduction of a narrative into the course further improved student engagement and attainment.

As future work, we are considering enhancements that can be made to the framework's story engine to make narratives more realistic and immersive, e.g. the addition of social media communication or pre-recorded videos.

Acknowledgement. This work was supported by grant GEN1214 from the Higher Education Academy.

References

1. Beuran, R., Chinen, K.i., Tan, Y., Shinoda, Y.: Towards effective cybersecurity education and training (2016)
2. British Board of Film Classification: `http://www.bbfc.co.uk`
3. Bursztein, E., Gourdin, B., Fabry, C., Bau, J., Rydstedt, G., Bojinov, H., Boneh, D., Mitchell, J.C.: Webseclab Security Education Workbench. In: 3rd Workshop on Cyber Security Experimentation and Test (CSET '10) (2010)
4. Carnegie Mellon University: picoCTF 2017. `https://2017.picoctf.com/about`
5. Chapman, P., Burket, J., Brumley, D.: PicoCTF: A Game-Based Computer Security Competition for High School Students. In: 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE '14) (2014)
6. Chothia, T., Holdcroft, S., Radu, A.I., Thomas, R.J.: Jail, Hero or Drug Lord? Turning a Cyber Security Course Into an 11 Week Choose Your Own Adventure Story. In: 2017 USENIX Workshop on Advances in Security Education (ASE 17). USENIX Association (2017)
7. Chothia, T., Novakovic, C.: An Offline Capture The Flag-Style Virtual Machine and an Assessment of Its Value for Cybersecurity Education. In: 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15) (2015)
8. Cyber Security Challenge UK: `https://www.cybersecuritychallenge.org.uk`
9. Davis, A., Leek, T., Zhivich, M., Gwinnup, K., Leonard, W.: The Fun and Future of CTF. In: 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE '14) (2014)
10. Entertainment Software Rating Board: `http://www.esrb.org`
11. Feng, W.: A "Divergent"-Themed CTF and Urban Race for Introducing Security and Cryptography. In: 2016 USENIX Workshop on Advances in Security Education (ASE 16). USENIX Association, Austin, TX (2016), `https://www.usenix.org/conference/ase16/workshop-program/presentation/feng`
12. Ford, V., Siraj, A., Haynes, A., Brown, E.: Capture the flag unplugged: an offline cyber competition. In: Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education. pp. 225–230. ACM (2017)
13. Hex-Rays SA: IDA: Freeware Version. `https://www.hex-rays.com/products/ida/support/download_freeware.shtml`
14. Higgins, R., Hartley, P., Skelton, A.: The Conscientious Consumer: Reconsidering the role of assessment feedback in student learning. In: Studies in Higher Education. vol. 27 (2002)

15. Kapp, K.M.: The Gamification of Learning and Instruction: Game-based Methods and Strategies for Training and Education. Pfeiffer & Company, 1st edn. (2012)
16. Logan, P.Y., Clarkson, A.: Teaching students to hack: Curriculum issues in information security. SIGCSE Bull. **37**(1), 157–161 (Feb 2005). https://doi.org/10.1145/1047124.1047405, `http://doi.acm.org/10.1145/1047124.1047405`
17. Massachusetts Institute of Technology, University of Cambridge: Cambridge 2 Cambridge. `https://cambridge2cambridge.csail.mit.edu`
18. McDaniel, L., Talvi, E., Hay, B.: Capture the flag as cyber security introduction. In: System Sciences (HICSS), 2016 49th Hawaii International Conference on. pp. 5479–5486. IEEE (2016)
19. Mirkovic, J., Peterson, P.: Class Capture-the-Flag exercises. In: 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE '14) (2014)
20. MWR InfoSecurity: HackFu: Run Your Own Event, Part 3: Creating the Theatrics. `https://hackfu.mwrinfosecurity.com/run-your-own-event/the-theatrics/index.html` (2017)
21. National Cyber Security Centre: CyberFirst courses. `https://www.ncsc.gov.uk/information/cyberfirst-courses`
22. Oracle Corporation: VirtualBox. `https://www.virtualbox.org`
23. Radu, A.I., Thomas, S.L.: Organising monkeys or how to run a hacking club. In: Workshop on Cybersecurity Training & Education (VIBRANT15) (2015)
24. Schreuders, Z.C., Ardern, L.: Generating randomised virtualised scenarios for ethical hacking and computer security education: SecGen implementation and deployment. In: Workshop on Cybersecurity Training & Education (VIBRANT15) (2015)
25. Sheldon, L.: The Multiplayer Classroom: Designing Coursework As a Game. Course Technology Press, Boston, MA, United States, 1st edn. (2011)
26. Stott, A., Neustaedter, C.: Analysis of gamification in education. Surrey, BC, Canada **8** (2013)
27. Trottier, D.: The screenwriter's bible: A complete guide to writing, formatting, and selling your script. Silman-James Press (1998)
28. University of Cambridge: Inter-ACE. `https://inter-ace.org`
29. Vigna, G., Borgolte, K., Corbetta, J., Doupé, A., Fratantonio, Y., Invernizzi, L., Kirat, D., Shoshitaishvili, Y.: Ten years of iCTF: The good, the bad, and the ugly. In: 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE '14) (2014)

A Sample Emails from our Story

As described in Section 6.2, the story engine advances the plot in part by sending emails from various characters in the story to an email account accessible to the student from within the VM. Two emails that form part of the story we created for our course are shown below; the student assumes the role of Employee 427.

A.1 Introduction of Chimp

The story begins with this email to Employee 427 from a mysterious character initially known only as Chimp.

```
Hey!

I told those guys in IT they need to give you stronger encryption keys
for email. Guess old moneybags decided it's too expensive to actually
care. What do you care, anyway? You're the new cybergeek I see - what a
generic term nowadays which has absolutely no context.

Who am I? You'll find out soon enough, but you need to prove youself to
me first. Why am I emailing you? Well, congratulations smarto - you
bagged last place in the prize list. The guy who sat in your seat was
involved in something big, but he went missing. So... what happens if
one of your best goes 'away'? You replace them with someone better, or
at least that's probably what HR said to you to sell the job.

This is where you come in. The email you just got from Adler? There's
more context than just a simple decryption task to get you started.
Working in 'cahoots' with the underworld is the manager's game, pinning
it on the little people in that bottom 99.99% leaves them grinning like
a cheshire cat. You had better know what I'm getting at or I'm finding
someone else.

So - those files you got for 'decryption'? Giving the answers to the top
0.001% isn't going to go down well for someone. Someone who is
completely innocent and has zero involvement, but they want to get rid
of soooo much.

All you have to do is give me as many cryptographic tokens you can find
inside them instead, and satisfy the idiots upstairs on the 42nd floor
by sending them some junk response a few minutes later - leave it til
your lunch break if you want. I don't really care how you play them off.

Anyway - I'm not going to tell you my life story, and I *really* don't
want to hear how your life story almost became some game. Just do what I
say and I'll make sure you're safe - just don't give me any curve balls,
and remember. *Once you're in, there's no leaving*.
```

```
  /~\
 C oo
  _( ^)
 / ~\
```

A.2 An Epilogue

This email concludes one of the paths through the story, and is sent from Nik to Employee 427 after the student completes the final exercise if they chose to work with Nik in the third stage of the plot (represented by the red path in Fig. 4).

```
427,
Your arraignment is looming - you’ve been arrested, charged and judge,
jury and (pity!) executioner are sending you for a little ’trip’.
We’ll be there with open arms as you are brought into your cell, only to
be known to the inmates as ’the traitor’.

Here’s something to think about before you arrive. Sleep with one eye
open. You’re mine now.

N.
```

B Emails Sent by Students to the Story Engine

As discussed in Section 7.2, 7 students displayed greater-than-average engagement with the story by sending (sometimes in-character) prose to the story engine along with the flags they had recovered. The story engine forwards copies of all emails sent to it by the student to our telemetry server (refer to Section 5.5 for further details); a sample of the emails these students sent is shown below.

- Hey mate, would you mind putting this key in your
  ~/.ssh/authorized_keys? No particular reason

- Found a token, have fun: 653d72c294c382de153dccce86f63ddb

- Hi there,
  Something big you say? I hope that I can trust you with these...

- Here.
  855e8fb63feed93e2c73785fc83737cf
  65e802467c57f7d0ecac094ad9d496af
  14673f7f3467e826b9f0425b5f14466a
  What is really going on in this place?
  427.

- Subject: HELP!!!
  I have some incriminating evidence on my bosses!
  I don't know who to turn to!
  Here's some statements from my boss' private directory!

- Well I'm just interested to see what happens here, I'll take the red pill.
  Here's the first token: 463325b2759dc7d7c901755c6876b187

C Post-Course Story Survey Feedback

To evaluate the impact of the introduction of a story to the 2016/17 iteration of the course (see Section 7.2), we surveyed students after the course had ended for their opinions on the story. We received 52 responses; a sample of the feedback the students gave is shown below.

- "I loved the story but it seemed to finish abruptly, and it wasn't long enough!
  "More emails would have been nice too, as we only got to interact with the story five times (one for each exercise).
  "I did like the complicated underground manoeuvres of the Sensible Furniture crowd. The Charles Garcia reveal and discovering the message that led to his demise was also a big moment in the tale.
  "RIP Chimp, may he never be forgotten"
- "Was genuinely upset when Chimp died. RIP."
- "Loved it. Great idea from start to finish!"
- "I found the story more enjoyable after finishing all of the exercises because then the story could be retried and different endings could be found."
- "I liked the opportunity to choose a path, but also be able to change at certain points. Felt involved with the characters and had a fitting ending."
- "The bad guys got what they deserved! Justice yay!"
- "Interesting to see how the story developed from certain situations."
- "I didn't lose (end up in jail), the taste of victory is sweet."
- "It's fun and enjoyable and definitely sets the exercises apart from other courses"
- "Gives context to the exercises, bit of fun to make people want to do them."
- "It gives the exercises meaning, rather than doing them for the sake of doing them"
- "Engages students to pay attention to the exercises, gives them a little real-life context (which often aids understanding) and instils confidence in students that the course is being very well-managed."