Design and Evaluation of Competition-based Hacking Exercises

Eman Alashwali
Information Systems Department
King Abdulaziz University
Jeddah, Saudi Arabia
ealashwali@kau.edu.sa

Hanene Ben-Abdallah
Information Systems Department
King Abdulaziz University
Jeddah, Saudi Arabia
hbenabdallah@kau.edu.sa

***Abstract*— This paper describes the design and delivery of two competitive-based small offensive security exercises in an undergraduate Computer and Information Security course at the Faculty of Computing and Information Technology, King Abdulaziz University. We designed competition scenarios for two small exercises based on known attacks. The first exercise aimed to break the Windows Server 2008 password, and the second sought to break the Wired Equivalent Privacy (WEP) wireless network key (password). We present the competition scenarios and design, including the required hardware and software in each exercise. In addition, we give an overview about the attacks and possible defenses against them. We also present the results of a survey conducted to determine students' sentiments towards these types of exercises and to measure the effectiveness of these exercises in supporting the course's theoretical concepts from the student perspective. The results strongly suggest that the exercises were informative, motivating, stimulating, and enjoyable. This work was only the first step for us. We look forward to creating more challenging competitive-based exercises and rewarding the teams that put forth superior efforts.**

Keywords*— Security; Computer hacking; Information security; Education; Computer science education; Engineering education

I. INTRODUCTION

With the growing number of cyber-attacks, information security education[1] is more important than ever before. Teaching students offensive techniques that allow them to think like an attacker (i.e., teaching them the security mindset) became essential to preparing good security engineers. In fact, in the information security community, it is not uncommon to hear that good security experts are good hackers, too.

As Schneier [1] describes it :

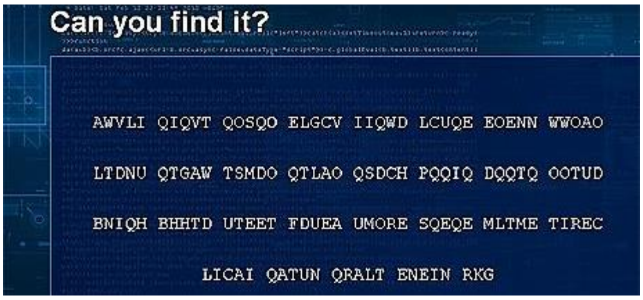
> *The security mindset involves thinking about how things can be made to fail. It involves thinking like an attacker, an adversary or a criminal.*

He continues to highlight the importance of the security mindset:

> *The lack of a security mindset explains a lot of bad security out there: voting machines, electronic payment cards, medical devices, ID cards, internet protocols... Teaching designers a security mindset will go a long way toward making future technological systems more secure.*

Hacking skills are extremely valuable these days, not only for security and intelligence agencies but also for Information Technology (IT) companies. For example, in 2013, the Government Communications Headquarters (GCHQ), a UK national security agency, launched a 4-stage online code breaking puzzle for recruitment purposes. The competition was titled "can you find it?" (See Figure 1). Only those who broke the code were asked for their contact information to be considered for a job at the GCHQ [2].

Valuing hacking skills as much as GCHQ does, most of the large IT companies like Mozilla, Microsoft, Google, Facebook, and Twitter appreciate and reward white hat hackers' efforts to report the security bugs in their systems [3] [4] [5] [6] [7]. For example, Microsoft launched the "Mitigation Bypass Bounty" program, which provides a reward of $100,000 USD to white hat hackers who can exploit security vulnerabilities in the latest version of the Microsoft operating system [4]. And in 2011, Facebook announced the "Security Bug Bounty" program [6]. Their bounties start at $500 USD and increase based on the severity of the bug reported [6]. In 2014, Facebook reported having paid more than $2 million in bounties since launching the program in 2011[8]. Furthermore, by 2013, two of the bounty recipients accepted full-time positions with the Facebook security team [9].



Can you find it?

AWVLI QIQVT QOSQO ELGCV IIQWD LCUQE EOENN WWOAO
LTDNU QTGAW TSMDO QTLAO QSDCH PQQIQ DQQTQ OOTUD
BNIQH BHHTD UTEET FDUEA UMORE SQEQE MLTME TIREC
LICAI QATUN QRALT ENEIN RKG

Fig. 1 CGHQ "Can you find it" recruitment competition [2].

[1] In this paper, by information security education, we mean technical education for university students.

The security mindset differs from the normal engineering mindset; in the normal engineering mindset, engineers think about how to make things work. In order to prepare students for the security mindset, they must be trained in or exposed to hands-on exercises that allow them to practice thinking like an attacker. In 2014, in an interview with Reginaldo Silva (the recipient of Facebook's largest single bounty worth $33,500 USD), Silva was asked: "What particular skills or traits do you think are required to be effective as a researcher?" He said [10]:

> *I'm always looking for the counterexamples. If a given assumption is valid 99 out of 100 times, I'm always trying to find the 1 time where the assumption is not valid.*

Interestingly, hacking does not always require sophisticated skills or advanced devices. A system can be hacked simply by thinking differently; that is the "security mindset". For example, in 2014, five-year-old Kristoffer Hassel was officially acknowledged by Microsoft in the *"Security Researcher Acknowledgments for Microsoft Online Services"* [11] for breaking the security of Microsoft Xbox and bypassing his father's account verification by typing spaces [12]. In 2013, there is the case of Khalil Shreateh, an unemployed information systems graduate from Palestine, who, with a "five-year-old laptop with broken keys and a broken battery", was able to exploit a Facebook bug that allowed him to post on any Facebook page outside his friends list [13]. After several failed attempts by Shreateh to get the Facebook security team's attention, he demonstrated the threat by hacking Mark Zuckerberg's personal Facebook page [13].

Undoubtedly, hands-on exercises similar to these real-life examples play a vital role in Computer Science and Engineering education. Good teachers always strive to design informative practical exercises that support the theoretical concepts presented in the lectures. They also work to capture students' interest, to motivate and stimulate them, and to provide them with enjoyable and competitive exercises.

In our previous work [14], we shared findings from an experiment that involved incorporating hacking projects into Computer and Information Security (CIS) lab exercises. The students' feedback revealed a positive response to hacking lab exercises. This paper presents an extension of our efforts at improving CIS lab exercises by teaching hacking skills and the security mindset via small competitive-based hacking exercises.

This paper has a two-fold motivation. First, we aim to improve the CIS lab exercises by designing and delivering competitive-based hacking exercises. The exercises were related to theoretical concepts covered during the lectures. Secondly, we aim to raise awareness of white hat hacking, ethical hacking, and security mindset skills, all of which refer to the same thing, namely: thinking like an attacker in order to find vulnerabilities before the bad guy does. We believe that such skills are not widely known, practiced, properly fostered, guided, or advanced in our region (Middle East) by educators or employers. An analogy to describe the importance of fostering and guiding hacking skills involves the art of graffiti. Graffiti can be a valuable nice piece of art if the talented graffitists find the right place to practice it, proper guidance, and appreciation; otherwise, it is considered harmful and may be an illegal offense.

The remainder of this paper is organized as follows: in section 2, we summarize several relevant works on hacking competitions; in section 3, we provide a brief background about the CIS course, the subject of this study; in section 4, we describe each exercise scenario and the competition design; in section 5, we present the evaluation results; and finally, in section 6, we conclude.

II. Background

In this section, we give a brief background about the CIS course at the Faculty of Computing and Information Technology, King Abdulaziz University (FCIT-KAU) - ladies campus, where the experiment was conducted.

The CIS course is a mandatory course in the three academic departments at FCIT: Computer Science (CS); Information Technology (IT); and Information Systems (IS). Originally, the lab curriculum of this course contains mostly Java programming exercises in which students develop small programs that implement cryptographic algorithms such as the Data Encryption Standard (DES), Rivest Shamir Adleman (RSA), etc. using Java libraries. In some primitive algorithms such as "Caesar" and "Vigenère" ciphers, the students are supposed to program the code without the use of Java libraries. The lab exercises did not contain hacking exercises or projects. Some breaking methods could have been mentioned in the lectures but to the best of our knowledge, students had never tried to implement an attack in either a project or a small exercise.

In the 2012/2013 academic year, we proposed incorporating hacking exercises into the lab curriculum. Our experiment presented in detail in *Incorporating hacking projects in computer and information security education: an empirical study* [14]. This paper is a continuation of our previous work in [14]; it is intended to improve the CIS lab and to offer exercises that help students develop a security mindset. In this work, we used a different method to deliver the hacking exercises: small competition-based exercises covering various security topics and attacks. In this way we were able to involve all students in implementing multiple hacking exercises. In contrast, in the project-based method that we used in [14], each group was asked to select one topic from various proposed topics and implement one attack.

III. Competition Exercise Design

To perform the competition-based lab exercises, we proceeded as follows: first, we asked the students to organize into groups of up to four students per group. Second, we created a scenario for each competition. Third, we set up the necessary platform for the two exercises of the competition. For the first exercise, we created a Windows Server 2008 virtual machine; for the second one, we configured a WEP wireless Access Point (AP). Fourth, since hacking exercises can be misused either intentionally or unintentionally, and to avoid any legal or ethical issues, it was important to take precautions before we allow students to practice the attacks.

Therefore, we gave a short presentation about the cyber security law in Saudi Arabia. Then, and before allowing students implement any attack, we asked them to a sign a pledge stating the rules that must be obeyed. The rules described in detail in our previous work in which we dedicated a section entitled *Handling the ethical and legal concerns in implementing hacking project* [14]. Fifth, for each exercise, we provided a lab session that described the attack, i.e., why it occurred. We did not perform the attack in the lab, instead, we left this to students as the lab exercise. We provided the students with lab hand-outs that contained a step-by-step guide on how to perform the attack in addition to providing them with the required software and virtual machine (by sharing them in a server). Finally, we asked the students to conduct the attack. Each exercise was given a duration of one week. At the end of the week, the students were required to submit the result (the victim's password) along with a proof of their work (screen shots in the first exercise and video recording in the second one). In addition, for the second exercise, the students were required to perform the attack live during a 2-hour lab session that we called "live-demo" for the attack.

In general, all groups that could break the password were considered winners. However, the group that achieved the attacker's goal first received special recognition by having their names posted on the lab instructor's board and announcing their names in the next lab.

In the following sections, we will describe each exercise in more detail. The attacks we used in the competitions are known attacks; there are plenty of online resources that describe how to launch them. To some extent, today, they can be considered general knowledge. However, we must acknowledge that we learned about the two attacks and how to perform them from lab sessions provided by Vasileios Giotsas at University College London (UCL) [19] [20].

A. Exercise 1: Breaking Windows Server 2008 Password

1) An overview of the attack

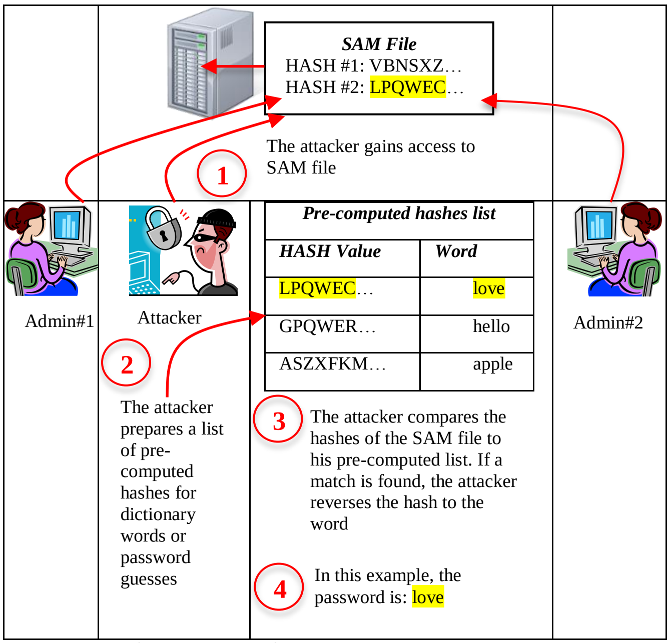
Generally, in order to authenticate local users using their passwords, operating systems store passwords locally (e.g. in a file). When users login, the typed password is compared to the stored one. Passwords must be stored securely so that it is impossible for anyone to deduce users' passwords if they accessed this file. In other words, passwords must not be stored in plain text.

In cryptography, a hash value is the output of a hash function, a function that takes an arbitrary size of data as input and outputs a fixed-size string. Cryptographic hash functions must fulfill some basic security properties. One of the most important properties is "pre-image resistance" which states that given a hash value *h*, it should be infeasible to find a message *x* such that *H(x) = h*. A hash function with this property is also called a "one-way" hash function, which means that it is infeasible to retrieve the original text simply by knowing the hash. Another important security property of hash functions is "collision resistance" which means it should be infeasible to find two messages *x*, *y* where *x ≠ y* such that *H(x) = H(y)*.

The above two properties of hash functions made them a suitable method for storing passwords for many operating system vendors, including Windows systems. When passwords are stored as hash values, the system authenticates the user by computing the hash of the entered password and compares it with the stored one.

However, storing passwords as hash values alone turned out to be insufficient: if the user chooses a weak password, an attacker who has access to the password hashes file (e.g., SAM file in Windows) can launch a brute-force or dictionary attack. The attacker computes the hashes of password guesses either by trying all possible combinations of digits or trying dictionary words and then comparing the resulted hash with the stored password hash until a match is found [21]. When a match is found, he reverses the hash to the original word which is known to him, and he finds the password [21]. Brute-force and dictionary attacks can be further improved using a rainbow-table attack in which the attacker prepares a list of pre-computed hashes [21]. Figure 2 shows a simplified illustration of the attack.

In our exercise, we used the Windows Server 2008 because it stores passwords as hash values only. It stores passwords using NT hash (which uses the MD4 hash algorithm) on the local disk in the Security Account Manager (SAM) file [22]. The previously described attacks can be easily launched if a user chooses a weak password and the attacker has access to the SAM file (further details in the attack scenario). Fortunately for attackers, by default, Windows systems (and may be other systems) allow users to create Administrators accounts with weak passwords or even with no passwords at all. The attack will be explained further in the following sub-sections (in which we address "what can go wrong").



The attacker gains access to SAM file

Pre-computed hashes list

HASH Value	*Word*
LPQWEC…	love
GPQWER…	hello
ASZXFKM…	apple

The attacker prepares a list of pre-computed hashes for dictionary words or password guesses

The attacker compares the hashes of the SAM file to his pre-computed list. If a match is found, the attacker reverses the hash to the word

In this example, the password is: love

Fig. 2 A simplified illustration for how a dictionary attack works. The rainbow table is improved technique.

2) The exercise's initial scenario

The initial state of the competition is as follows: we have two systems administrators: *Attacker1* and *Attacker2,* who share one Windows Server 2008 machine. Each administrator is a member of the Windows "Administrators" group which grants the administrator account full control over the system. Presumably, each administrator has his own Windows profile configured with the appropriate file permissions that protect his profile from being accessed by anyone other than himself. This scenario is realistic and can be found in some organizations where two employees share the same physical machine. It does not need to be a server; the same principle applies to desktop machines.

3) What can go wrong?

Since the passwords of Windows Server 2008 are stored only as hash values in the SAM file (i.e., they are not salted hashes), this makes the weak passwords (such as dictionary word passwords) prone to the rainbow-table attack. A dishonest administrator who has legitimate access to the system can exploit this fact and perform this attack on the stored hashes to retrieve other administrators' passwords (if they are weak). With the use of some free online tools, the victim's administrator(s) passwords can be found in a matter of seconds.

For simplicity, in this exercise, we assumed that the attacker is an internal attacker (i.e., a dishonest administrator who has an account in the system). However, similar attacks that exploit the password hashes can still be performed without the existence of an account for the attacker; all that's necessary is physical access to boot the machine from a Linux Live CD as shown in Computer Security Lab Session: Password Cracking by V. Giotsas [19].

Once the password is in the wrong hands, the entire security of the system is broken (i.e. confidentiality, integrity, and authenticity). As a result, the attacker can perform several types of attacks, including but not limited to the following:

- Masquerading attack: in which the attacker fakes his identity to perform operations under the name of the legitimate administrator.
- Denial of service attack: in which the attacker denies the legitimate administrator access to the system by, for example, changing the legitimate administrator's password.
- Modification attack: in which the attacker adds/modifies/deletes data without the legitimate administrator's consent or knowledge.

4) The competition design

a) Competition setup: to set up the competition, we installed the Windows Server 2008 operating system on a virtual machine. Using virtual machines has several advantages including providing a secure training environment in which our exercise attempts are isolated from the actual machine so that we do not compromise a real system's security. In addition, virtual machines provide simplicity: the exercise required one installation for the operating system, and then the virtual machine could be copied to the students. Furthermore, virtual machines provide portability; the virtual machine can be run on any computer.

In our virtual machine, we created three initial Administrator accounts with initial passwords. The three accounts were as follows:

- CPIS312: Administrator account. To be used by both groups to reset their group's administrator password.
- Attacker1: Administrator account. To be used by the first competing group (Group#1).
- Attacker2: Administrator account. To be used by the second competing group (Group#2).

Before the students begin, they must organize into groups of up to four students. Each group must find a competitor group to share the Windows Server machine with.

Next, we provided a copy of the exercise's virtual machine to every pair of competing groups to be used in this exercise. At the beginning of the exercise, every pair of competing groups must set up their shared machine together. Each group must decide which Administrator account represents them (either *Attacker1* or *Attacker2*). Then, each group secretly chooses a password for their Administrator and resets the initial password to the password they chose. The password must be a weak password: a lower case dictionary word with maximum length of 6 characters. The chosen password must remain secret and must not be given to anyone outside the group (except the instructor).

After each group set their Administrator password, they were required to fill out a form for the instructor that included their chosen password (as shown in Figure 3).

In the next stage, each group took a copy of the virtual machine (that they had configured with their competitor) and starting from this stage the two groups work separately. Each group tried to break their competitor's password first. Hence, each group played two roles: the attacker and the victim roles.

b) The software and hardware needed: in order to perform the attack, we needed the following software:

- VMware workstation: a commercial software to create virtual machines and host various operating systems [23]. We are considering using free alternative software in the future.
- The ISO files for Windows Server 2008: can be obtained from either the trial version or from other free sources for educational purposes such as Microsoft DreamSpark [24]
- Ophcrack: free software by which we can load rainbow-tables to break passwords [25]
- The rainbow tables for all lowercase words: there are various tables but we used this for our exercise. It is free and available at [26].
- Cain & Abel: free software that allowed us to extract the hashes from the SAM file [27].

- In terms of hardware, the students used their ordinary laptops to host the virtual machine and implement the attack.

c) The attacker's goal (winning the competition): a successful attack results in recovering the competitor's administrator account password in plain text as shown in Figure 4 below.

5) Defenses against dictionary and rainbow-table attacks

To defend systems against brute-force, dictionary, and rainbow-table attacks, systems should employ salted hashes, i.e., passwords that are combined with other random values unknown to the attacker. The system uses these combinations to compute the hashes [21]. This makes it infeasible for the attacker to prepare a pre-computed list of hashes [21].

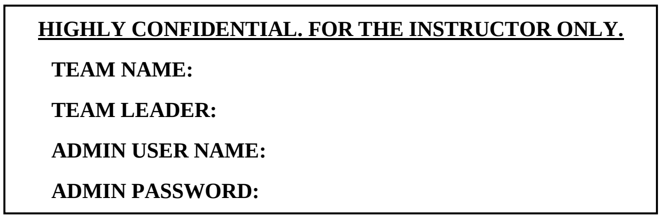
Most importantly, users must be educated to never create an account with empty or weak passwords. Users' passwords must meet complexity requirements [28] [29]. Password strength is based on two factors: the length and the character set. The longer the password and the larger the character set from which it is drawn, the more resistant to guessing and brute-force attacks. Passwords must not contain personal information that can be easily guessed; must not be dictionary words; and must contain at least three of the following categories: upper case letters; lower case letters; numbers (0-9); special characters; Unicode characters [28] [29].

B. *Exercise 2: Breaking WEP Wireless Network Password*

1) An overview of the attack

Wired Equivalent Privacy (WEP) is a security protocol introduced in 1997 as an IEEE standard [30]. It aimed to provide the security equivalent of the wired network, in particular [31]:

- Confidentiality: to protect against eavesdropping
- Authentication: to prevent unauthorized access
- Integrity: to prevent data modification



HIGHLY CONFIDENTIAL. FOR THE INSTRUCTOR ONLY.

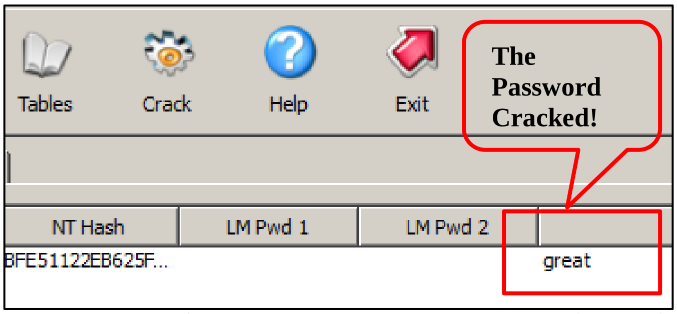
TEAM NAME:

TEAM LEADER:

ADMIN USER NAME:

ADMIN PASSWORD:

Fig. 3 The secret form submitted to the instructor by each group prior to performing the attack.



NT Hash	LM Pwd 1	LM Pwd 2	
BFE51122EB625F...			great

Fig. 4 Screen shot of the Ophcrack output showing the password "great" in plain text after it was cracked.

However, this protocol was shown to have serious design flaws and was officially deprecated in 2004 by IEEE. One of the main problems with this protocol was in the key generation process. WEP is designed to encrypt/decrypt data using a stream cipher. In stream ciphers, the cipher text is generated simply by XORing the plain text with a pseudo-random key stream. The decryption process works in reverse, by XORing the cipher with the same pseudo-random key stream. WEP protocol employs the RC4 stream cipher that uses the WEP key to generate the pseudo-random infinite key stream [31].

In stream ciphers, the same key must never be used twice [31]. It must be randomized. However, WEP is designed with a 64-bit key consisting of a 40-bit shared key concatenated with a 24-bit initialization vector (IV) [31]. There is another version of WEP that uses a 128-bit key that consists of a 104-bit shared key concatenated with a 24-bit IV. However, even the long-key version of WEP suffered from the same problems as the 64-bit version. We will use the 64-bit version in our exercise. Normally, in WEP, the 40-bit shared key is fixed and rarely changed, therefore, the key stream randomization relies only on the 24-bit IV value [31]. Due to the insufficient length of the IV, the key stream will definitely be repeated after at most 2^24 ≈ 16 Million different IVs are generated (i.e., frames) [31]. Even worse, the IV is sent in clear text, which helps the attacker know when the IV is repeated [31].

Due to the above design flaws, several attacks were possible. One of the attacks is the "two-time pad" attack, which occurs when two ciphers get encrypted with the same key [31]. To illustrate, since $C=P\oplus K$ (where C is the cipher text, P is the plain text, and K is the Key), when the key is reused, XORing two ciphers will give the result of XORing the two plain texts, i.e. $C1\oplus C2=P1\oplus P2$ [31]. When the attacker obtains $P1\oplus P2$, this gives him clues about the plain text which may allow him to obtain the plain text [31]. Normally, knowing the XOR of two plain texts is enough to recover both of the texts [31]. Another attack called "related-key" attack, which resulted from the fact that they keys are related to each other (fixed shared 40-bits and the 24-bit IV increments sequentially), can allow the attacker to extract the secret key (WEP key) by collecting and analyzing enough IVs (≈1 Million frames) [32]. This attack was first described by Fluhrer et al. in 2001 and known as the Fluhrer, Mantin, and Shamir (FMS) attack [33]. In 2007, the attack was improved by Tews et al. Their approach, which is known as Pyshkin, Tews, Weinmann (PTW), has reduced the number of needed IVs and hence the time to break the key [34]. The PTW method is the default method used in the free tool "Aircrack-ng" which we used in our exercise [35].

2) The exercise's initial scenario

The initial scenario is that there is a wireless AP configured with the flawed WEP protocol. The WEP password is secret and not known to the network users (i.e., the students). WEP is still found and provided in most of the APs and routers probably for backward compatibility. Therefore, un-aware end users may choose this broken protocol while configuring the security settings of their device. Unfortunately, this scenario still occurs in real-life cases.

3) *What can go wrong?*

Since anyone can easily discover the security protocol used in an AP either through the client's network settings or by using some network scanning tools, an attacker can target a WEP network. The result will allow him to retrieve the secret key. This permits him to perform different types of attacks including denial of service and traffic decryption to reveal encrypted data.

Breaking the WEP key can be done by an amateur in a matter of seconds using free tools available online.

4) *The competition design*

a) Competition setup: we connected a wireless AP. Then, we configured the AP with:

- Service Set Identification (SSID) (i.e., network name). We used: *CPIS312* as a network name.
- Security protocol. We selected the following password: *"group"* as the WEP key. We used a WEP 64-bit key.

Next, we provided the students with the network name. The password was kept secret and known only to the lab instructor. We ran two Aps, each one placed on a separate floor. The students were allowed one week to exercise and submit the answer (the key). In this competition, all student groups were playing the attacker role, competing with the instructor (the victim and the network owner).

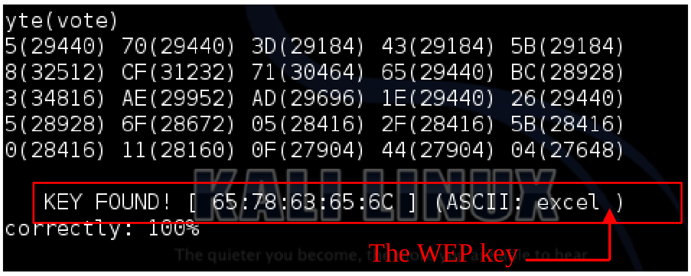
b) The software and hardware needed: in order to perform the attack, we needed the following software:

- Kali ISO image: Kali is a Linux distribution designed for penetration testing. It contains several penetration software including *"Aircrack-ng"* which we used for cracking the WEP network [36].

In terms of hardware, we needed:

- A wireless AP: any off-the-shelf AP that supports WEP protocol can serve the purpose.
- Wireless cards that are capable of packet injection. We used an Alfa wireless adapter [37].
- An ordinary laptop to host the virtual machine of Kali system.

c) The attacker's goal (to win the competition): a successful attack should extract the WEP key in plain text as shown in Figure 5.



```
yte(vote)
5(29440) 70(29440) 3D(29184) 43(29184) 5B(29184)
8(32512) CF(31232) 71(30464) 65(29440) BC(28928)
3(34816) AE(29952) AD(29696) 1E(29440) 26(29440)
5(28928) 6F(28672) 05(28416) 2F(28416) 5B(28416)
0(28416) 11(28160) 0F(27904) 44(27904) 04(27648)

   KEY FOUND! [ 65:78:63:65:6C ] (ASCII: excel )
correctly: 100%
```

Fig.5 Screenshot of the *Aircrack-ng* showing the password "*excel*" in plain text after it was cracked.

5) *Defenses against a WEP attack*

To defend against this attack, network administrators for home or enterprise networks must never use a WEP protocol to secure their networks. Instead, they should use improved protocols such as WPA2, which has resolved the problems in WEP. However, even when WPA2 protocols are used, attention must be paid to the password strength in order to defend against dictionary attacks [38].

IV. EVALUATIONS OF THE EXERCISES

In order to evaluate the exercises from the student's perspective, we conducted a survey.

A. *Sample*

Our sample consisted of 46 undergraduate female students, who represent all the students officially enrolled in the CIS course in the IS department in the second semester of the 2013/2014 academic year. The students' ages ranged from 20-24 years.

B. *Methodology*

At the end of the semester, after we finished all the lab exercises, we distributed an anonymous paper-based survey to the subjects. All the students returned the survey. Originally, 47 surveys were returned to us, due to one additional student who mistakenly joined and filled out the survey while she was not enrolled in the course (the student informed us that she did not complete the survey). Because the surveys were anonymous, we could not recognize her survey and, therefore, decided to exclude the most incomplete survey, which presumably belonged to this student.

We ended up with 46 filled surveys, two of which did not have all questions answered. However, we believe this does not affect the accuracy of the results, as we computed a weighted average and the results tables below show the total number of answers received for every particular question.

We used a 5-point Likert scale for this survey, with questions that measured the students' opinions. The survey included one yes/no question and two multiple choice questions. In addition, it included a section for evaluating the instructor (We did not include that section in this paper as it is beyond the scope of the paper). Table 1 in the Appendices section summarizes the survey questions and the answers provided.

C. *Results and Discussion*

In this section, we summarize our results. In the summary, for brevity, when we say "Agree" we refer to both "Strongly Agree" and "Agree"; the same is true for "Disagree." Detailed statistics can be found in the tables below.

The results indicate that 84.78% of the students agree that the lab exercise increased their knowledge regarding information security. The results also suggest that the exercises helped students better understand the theoretical concepts covered in the lectures, as 86.96% agreed with this assessment. In addition, 65.22% of the students believed that it would be difficult for them to understand how attackers think without practicing offensive security lab exercises.

The competitive-based exercise design appeared to be successful in motivating and stimulating the students, as 69.57% agreed with this assessment. In addition, 64.44% agreed that the exercises augmented the positive competitive spirit between the teams and 78.26% agreed that working on competitive-based exercises was enjoyable. Furthermore, 76.09% recommend competition-based offensive exercises to future students. In general, 65.22% of the students agreed that the offensive lab exercises met their expectations and 60.87% wished that there were more offensive security lab exercises given during the lab sessions throughout the semester. In addition, 50% wish that more grades were allocated to the offensive security lab exercises.

In terms of the difficulty of the exercises, we found that 58.70% assessed the level as “just right,” 23.91% found them “easy” or ”very easy,” while 17.39% believed that the exercises were “difficult” or ”very difficult.” As for the amount of time provided to practice the exercises and to submit the answers, 45.65% agreed that the time provided to complete the offensive exercises was adequate, 39.13% felt neutral about the amount of time, and 15.22% disagreed that the time provided was adequate. For the live demo exercise, which was a sort of examination of breaking the WPA wireless network live in front of the lab lecturer in about 1.5 hour time, 65.22% agreed that the amount of time provided was adequate, 13.04% disagreed, and 21.74% felt neutral about the amount of time provided.

V. Conclusion

In this paper, we presented the competitive-based hacking exercises that we designed and delivered to undergraduate students taking the CIS course. We described the set-up of each exercise and the way in which we designed the small competitions.

We evaluated the experiment from the student’s perspective. The survey results showed a positive attitude toward these types of exercises. The majority of the students found them to be informative, competitive, and enjoyable, and they recommend these exercises to future students.

This is our first attempt at conducting such in-class competitions. We look forward to increasing the number of these exercises and to specifying monetary rewards for winners. We will also consider giving the students more time. In addition, we plan to share the results of our experiment with the curricular committee at FCIT-KAU to use as an example and to argue the need for generalizing competitive-based exercises in CIS courses for the other departments in the college.

VI. References

[1] B. Schneier (2008, Mar. 25). Schneier on Security: The Security Mindset [Online]. Available: https://www.schneier.com/blog/archives/2008/03/the_security_mi_1.htm [Accessed: Sep. 9, 2014].

[2] A. Philipson (2013, Sep. 11). Can you crack the code? GCHQ unveils fiendish puzzle for new recruits [Online]. Available: http://www.telegraph.co.uk/news/uknews/defence/10301435/Can-you-crack-the-code-GCHQ-unveils-fiendish-puzzle-for-new-recruits.html [Accessed: Sep. 5, 2014].

[3] Mozilla (2013, MAy 22). Mozilla Security Bug Bounty Program [Online]. Available: https://www.mozilla.org/security/bug-bounty.html [Accessed: Sep. 8, 2014].

[4] The BlueHat team (2014). Microsoft Bounty Programs [Online]. Available: http://technet.microsoft.com/en-us/security/dn425036 [Accessed: Sep. 8, 2014]

[5] Google. Program Rules- Application Security - Google [Online]. Available: http://www.google.com/about/appsecurity/reward-program/ [Accessed: Sep. 8, 2014].

[6] Facebook (2014). Information [Oline]. Available: https://www.facebook.com/whitehat/ [Accessed: Sep. 8, 2014].

[7] HackerOne. HackerOne - Twitter [Online]. Available: https://hackerone.com/twitter [Accessed: Sep. 8, 2014].

[8] Facebook (2013, Apr. 3). Bug Bounty Highlights and Updates [Online]. Available: https://www.facebook.com/notes/facebook-bug-bounty/bug-bounty-highlights-and-updates/818902394790655 [Accessed: Sep. 29, 2014].

[9] Facebook (2013, Aug. 2). An update on our Bug Bounty Program [Online]. Available: https://www.facebook.com/notes/facebook-security/an-update-on-our-bug-bounty-program/10151508163265766# [Accessed: Sep. 29, 2014].

[10] Caseyjohnellis (2014, Feb. 5). Interview: Reginaldo Silva – Largest Facebook bug bounty awarded researcher [Online]. Available: https://blog.bugcrowd.com/reginaldo-silva-facebook-bug-bounty-awarded-researcher/ [Accessed: Sep. 8, 2014].

[11] Microsoft (2014). Security Researcher Acknowledgments for Microsoft Online Services - Previous Months [Online]. Available: http://technet.microsoft.com/en-us/security/cc308575 [Accessed: Nov. 17, 2014].

[12] M. Chen (2014, Apr. 7). 5-year-old Ocean Beach boy exposes Microsoft Xbox vulnerability [Online]. Available: http://www.10news.com/news/5-year-old-ocean-beach-exposes-microsoft-xbox-vulnerability [Accessed: Nov. 17, 2014].

[13] D. Gross (2013, Aug. 20). Zuckerberg's Facebook page hacked to prove security flaw [Online]. Available: http://edition.cnn.com/2013/08/19/tech/social-media/zuckerberg-facebook-hack/ [Accessed: Nov. 17, 2014].

[14] E. Alashwali, “Incorporating hacking projects in computer and information security education: an empirical study,” Int. J. Electronic Security and Digital Forensics, vol. 6, no. 3, pp. 185–203, 2014.

[15] The UCSB iCTF [Online]. Available: http://ictf.cs.ucsb.edu/#/ [Accessed: Nov. 1, 2014].

[16] C. Lee, A. Uluagac, K. Fairbanks, and J. Copeland, "The Design of NetSecLab: A Small Competition-Based Network Security Lab," IEEE Transactions on Educ., vol. 54, no. 1, pp. 149-155 Feb. 2011.

[17] Cyber Security Challenge (2014). Universally Challenged [Online]. Available: http://cybersecuritychallenge.org.uk/universally-challenged/ [Accessed: Nov. 14, 2014].

[18] Cyber Security Challenge (2014). Universally-Challenged-Description-2014-2015-1 [Online]. Available: https://cybersecuritychallenge.org.uk/wp-content/uploads/2014/08/Universally-Challenged-Description-2014-2015-1.pdf [Accessed: Nov. 1, 2014].

[19] V. Giotsas (2011). "COMPUTER SECURITY LAB SESSION : Password Cracking," University College London (UCL), unpublished.

[20] V. Giotsas (2011). "COMPUTER SECURIYT LAB SESSION 5: Wireless Password Cracking," University College London (UCL), unpublished.

[21] J. Ullrich (2011, Jun. 28). Hashing Passwords [Online]. Available: http://www.dshield.org/diary/Hashing+Passwords/11110 [Accessed: Nov. 11, 2014].

[22] Microsoft (2013, Sep. 12). Cached and Stored Credentials Technical Overview [Online]. Available: http://technet.microsoft.com/en-us/library/hh994565.aspx [Accessed: Nov. 11, 2014].

[23] Vmware (2014). Vmware Workstation [Online]. Available: http://www.vmware.com/products/workstation [Accessed: September 01, 2014].

[24] Microsoft (2014). Microsoft DreamSpark - Software Catalog [Online]. Available: https://www.dreamspark.com/student/software-catalog.aspx [Accessed: Nov. 15, 2014].

[25] Ophcrack (2014). Ophcrack [Online]. Available: http://ophcrack.sourceforge.net/ [Accessed: Nov. 21, 2014].

[26] Sourceforge (2014), Browse /tables/Vista free at Sourceforge.net [Online]. Available: http://sourceforge.net/projects/ophcrack/files/tables/Vista%20free/ [Accessed: 02 June 2014].

[27] Oxid.it (2014. Cain & Abel [Online]. Available: http://www.oxid.it/cain.html [Accessed: Nov. 21, 2014].

[28] Microsoft (2014). Password Strength - Password Strength Calculator and Password Checker [Online]. Available: https://www.microsoft.com/security/pc-security/password-checker.aspx [Accessed: Nov. 16, 2014].

[29] US-CERT (2014, Feb. 6). Choosing and Protecting Passwords [Online]. Available: https://www.us-cert.gov/ncas/tips/ST04-002 [Accessed: Nov. 16, 2014].

[30] IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE Std 802.11-1997, pp. 1- 445, 1997.

[31] I. Goldberg, N. Borisov, D. Wagner (2001, Jul. 1). The Insecurity of 802.11 An analysis of the Wired Equivalent Privacy protocol [Online]. Available: http://www.cypherpunks.ca/bh2001/mgp00001.html [Accessed: Nov. 12, 2014].

[32] D. Boneh (2014). Attacks on stream ciphers and the one time pad. coursera.com [Online]. Available: https://class.coursera.org/crypto-010/lecture/6 [Accessed: Nov. 14, 2014].

[33] S. Fluhrer, I. Mantin and A. Shamir. "Weaknesses in the Key Scheduling Algorithm of RC4," in Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography (SAC'01), Serge Vaudenay and Amr M. Youssef (Eds.). London, UK, 2001, pp. 1-24.

[34] E. Tews, R.-P. Weinmann, and A. Pyshkin. "Breaking 104 bit wep in less than 60 seconds," in Proceedings of the 8th international conference on Information security applications (WISA'07), Berlin, Heidelberg, 2007, pp. 188-202.

[35] Darkaudax (2013, Feb. 3). aircrack-ng [Online]. Available: http://www.aircrack-ng.org/doku.php?id=aircrack-ng [Accessed: Nov. 14, 2014].

[36] Offensive Security Ltd. (2014). Kali Linux [Online]. Available: http://www.kali.org/ [Accessed: Sep. 4, 2014].

[37] ALFA NETWORK Inc. (2011). 802.11b/g/n Long-Rang USB Adapter [Online]. Available: http://www.alfa.com.tw/product_category.php?pc=3 [Accessed: Sep. 4, 2014].

[38] Mister_x (2010, Mar. 7). Tutorial: How to Crack WPA/WPA2 [Online]. Available: http://www.aircrack-ng.org/doku.php?id=cracking_wpa [Accessed: Nov. 17, 2014]

VII. Appendices

TABLE I. Our survey results (1)

Q. No.	*Question*	**Answers**					
		Strongly Agree	***Agree***	***Neutral***	***Disagree***	***Strongly Disagree***	***Total Answers***
1	The offensive security lab exercises increased my knowledge about information security	21 (45.65%)	18 (39.13%)	6 (13.04%)	0 (0.00%)	1 (2.17%)	46
2	The offensive security lab exercises helped me better understand the theoretical concepts covered in the lectures	14 (30.43%)	26 (56.52%)	5 (10.87%)	0 (0.00%)	1 (2.17%)	46
3	It will be difficult for me to understand how attackers think without practicing the offensive security lab exercises	15 (32.61%)	15 (32.61%)	11 (23.91%)	5 (10.87%)	0 (0.00%)	46
4	Working on competition-based (i.e. against another student group or against the instructor) offensive security lab exercises was motivating and stimulating	18 (39.13%)	14 (30.43%)	13 (28.26%)	0 (0.00%)	1 (2.17%)	46
5	Working on competition-based offensive security lab exercises augmented the positive competitive spirit between the groups	18 (40.00%)	11 (24.44%)	15 (33.33%)	0 (0.00%)	1 (2.22%)	45
6	Working on offensive security lab exercises in competition-based method was enjoyable	17 (36.96%)	19 (41.30%)	7 (15.22%)	2 (4.35%)	1 (2.17%)	46
7	I recommend teaching the offensive security lab exercises in a competition-based method to future students	16 (34.78%)	19 (41.30%)	7 (15.22%)	2 (4.35%)	2 (4.35%)	46
8	The time provided to practice the offensive security lab exercises was adequate	8 (17.39%)	13 (28.26%)	18 (39.13%)	7 (15.22%)	0 (0.00%)	46
9	The time provided to accomplish the “Live Demo” lab objectives was adequate	11 (23.91%)	19 (41.30%)	10 (21.74%)	4 (8.70%)	2 (4.35%)	46
10	The number of students per group was adequate	19 (42.22%)	18 (40.00%)	6 (13.33%)	2 (4.44%)	0 (0.00%)	45
11	The offensive security lab exercises met my expectations	10 (21.74%)	20 (43.48%)	14 (30.43%)	1 (2.17%)	1 (2.17%)	46
12	I wish there were more offensive security lab exercises given in the lab	14 (30.43%)	14 (30.43%)	10 (21.74%)	7 (15.22%)	1 (2.17%)	46
13	I wish there were more grades allocated to the offensive security lab exercises (currently 8 grades)	10 (21.74%)	13 (28.26%)	14 (30.43%)	9 (19.57%)	0 (0.00%)	46

TABLE I. THE LEVEL OF EXERCISES DIFFICULTY FROM STUDENTS PERSPECTIVE

Q. No.	***Question***	***Strongly Agree***					
		Very Difficult	***Difficult***	***Just Right***	***Easy***	***Very Easy***	***Total Answers***
14	The level of difficulty of the offensive security lab exercises was adequate	1 (2.17%)	7 (15.22%)	27 (58.70%)	8 (17.39%)	3 (6.52%)	46