Using *Capture the Flag* in Classroom: Game-based Implementation in Network Security Learning

Harsa Wara Prabawa, Enjun Junaeti, Yana Permana
Universitas Pendidikan Indonesia
Bandung, Indonesia
harsawara@upi.edu, enjun@upi.edu, yana.permana@student.upi.edu

***Abstract*—The development of Information and Communication Technology (ICT) significantly affects life in general. On the contrary of the dynamics development of security science and high demand of knowledge and practical ability to handle the network and information security problems; network security learning is taught with theoretical dominance– the methods tend to be passive in nature for students and uses traditional approaches that are through lectures and textbooks. The study attempts to discuss technology intervention in network security learning in form of media learning named Capture The Flag game – with little involvement of Boyer-Moore Algorithm for answer-matching calculation. To test the media appropriateness, the study implements experiment in one of vocational school of Network and Computer Engineering in Bandung. By using LORI instrument, the experimental results show that the developed media is categorized as appropriate to be implemented and used as learning guidelines.**

Keywords—capture the flag; game-based learning;boyer-moore algorithm

I. Introduction

Recent Information Communication and Technology (ICT) development plays significant role for life- some of which are utilized in real activities that bring positive benefits. However, the emergence of technology inevitably opens space to facilitate the crime, especially cybercrime which certainly involves particular computer network. At the same time, the user number of unsophisticated computer and vulnerable PC, which always connects to the internet, is growing rapidly. Therefore, it is normal when many jobs accentuate experiences and practical skill as the keys for job requirement, especially for security professionals; and those should be noteworthy part of security course.

Unlike the dynamics of security scientific development and the high demand for knowledge and practical ability to handle the network and information security problem, network security learning is taught merely with theoretical dominance. The method tends to be passive for students- some kind of traditional approach that is through lectures and textbooks [1] [2]. Indeed, network security materials contain both theory and practice. The harmony between practice and theory is required so that the students fully understand the theory; and it enables to improve their quality of understanding. Learning practice plays paramount roles, not only in verifying the theories students have gained, but also their result of hard work. More importantly, it helps in improving students' professionality as well as forming inspiring and innovative students through the process of imitating, organizing and understanding; until they are capable to analyze independently and completely the entire training process.

Capture the Flag (CTF) is a type of game specifically designed for the field of information and network security [3]. CTF originally starts as traditional game, which is played manually by two groups of players. Each group of players is obliged to move the opponent's flag to the group's own defense side. The opposing group must certainly try as much as possible to defend the group's flag so as not to be seized by the opponent. In its digital era implementation, CTF evolves into a network and information security game [3], even used as a teaching and learning material [4], and as well as a selection tool in network and information security competitions [5]. As part of the competition selection, CTF is a game that encourages participants to think outside the box. In this game, participants are confronted by several issues to be solved. The problems are defined in terms of levels. Thus, the higher the level achieved by the participants, the closer the participants reach the end of the CTF game.

The study attempts to present technological intervention attempt in network security learning in the form of CTF game learning media . CTF, as part of laboratoty practices, provides Hands-on Labs that plays key role in introducing networking concepts and network security skills as those can assist students to apply basic safety principles and techniques for the protection of computers and real-world network systems.

II. Capture the Flag as a Learning Media

Generally, CTF game is divided into three-frequently-used-scenarios: Jeopardy, Attack-Defense and Mixed [6]. In Jeopardy scenario, the players are asked to solve several tasks – to gain and collect points. The winner is the player who has the highest point accumulation. In Attack-defense scenario, the players are divided in to two teams. Each team play the role to attack and maintain the computer system provided to the team. Both teams are obliged to attack the opponent team as well as protect their asset. More precisely, each team is responsible to protect a host set and withhold the flag (confidential information) within the host. Each team is not

978-1-5090-5864-8/17/$31.00 ©2017 IEEE

allowed to prevent another team to attack their host. Instead, the team's priority is to detect attacks from opponents. Additionally, each team must attack another team and retrieve the flags for each hacked host. Mixed scenario is implied as the mixture of the two previous scenarios [3].

The study adapts CTF jeopardy scenario due to the learning objective which is students' understanding towards network security materials. Network security materials provided in CTF game are presented in problem based learning scenario. The materials are subsequently developed as a challenge in form of possibly various treats and attacks to network security. Generally, learning scenario designed on CTF network security learning media are presented on Fig. 1.

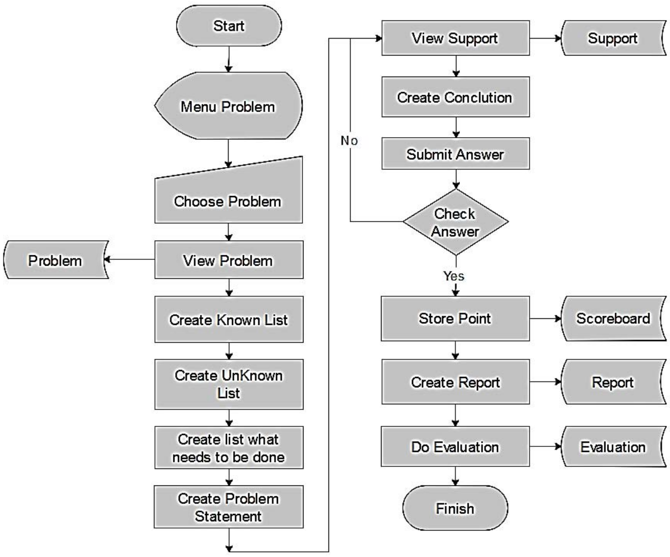
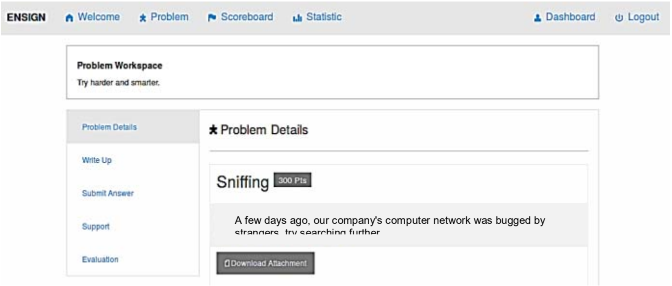


Fig. 1. CTF learning scenario.

On the first stage, the students select one of problems provided in network security learning media. For instance, the students choose a problem entitled Sniffing, once they click, the Problem Workspace will appear as displayed in Fig. 2.



Problem Workspace

Try harder and smarter.

★ Problem Details

Sniffing 300 Pts

A few days ago, our company's computer network was bugged by strangers, try searching further

Fig. 2. CTF problem workspace.

The students read at glance about the problems and download the necessary files which are required to be analyzed. The files are PCAP file which are utilized to record network footprint and other data streams (Fig. 3).

On this stage, the students enable to open Write-up tab to complete the information collected by them. For example, *Known List*, the student can fill about the information they gained from the network footprints record.

Known List: *Protocols provided are TCP, ARP, BROWSER, etc. As for each network packet has its own communication, for example in TCP protocol, the source of IP address is 192.168.56.100 and the destination of IP address is 192.168.56.1.*

Further, the students enable to continue the progress by completing *Unknown List*, for example;

Unknown list: *confidential communication between two inter correlated entities; sensitive and confidential information such as password, credit card PIN and much more.*

In this multimedia, the sensitive information is considered as flag. The sequential step is the students fill the list with the action they can implement after discovering the available information.

To do list: *reading every conversations or important IP communication and soliciting for the confidential information named flag*

In Problem Statement tab, the student can fill it with the summary of recent encountered problems, for instance:

Problem statement: *The recent case we encounter is Sniffing with various protocols, such as TCP, ARP BROWSER, etc.*

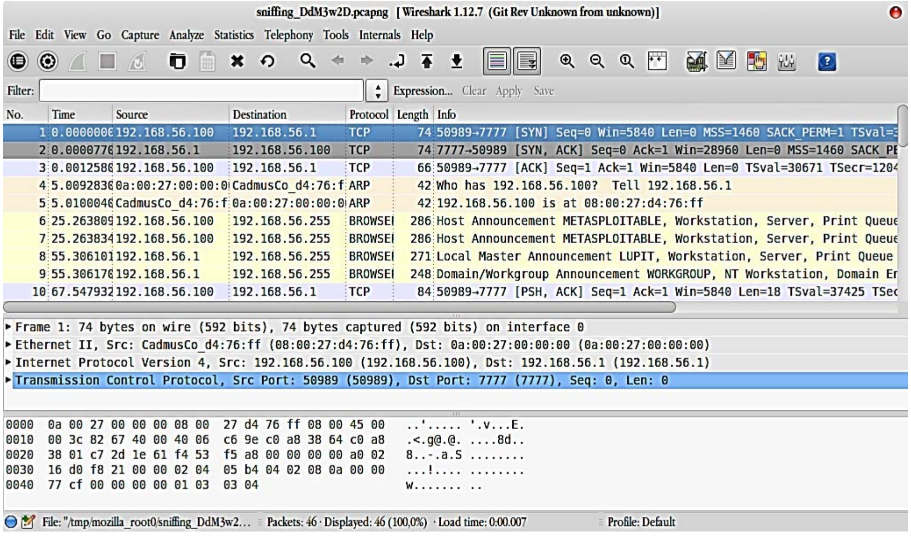
The students subsequently implement things, which have been defined in To Do List tab, such as reading every conversation. For example, the students attempt to see one of TCP packet conversation and anything associated with it. The students try to read the available information, for instance they try to inspect the packet by Expand Tab the Wireshark on each TCP/IP layer (Fig. 4).

If the other confidential information has not discovered yet (flag), the students can use the available filter in Wireshark (Fig. 5). For instance, the students only want to filter the packet with TCP protocol.

The students enable to check the content of each TCP/IP layer by implementing the previous method. If the information has not been disclosed (flag), they can try other Wireshark filters by using 'tcp and ip.src eq 192.168.56.100'

At this stage, the students are permitted to use the `Follow TCP Stream` feature, which is used to string all the packets into full conversations. For example, the students try to check the conversations or communications between IP 192.168.56.100 and 192.168.56.1. The results of confidential information successfully discovered by the students are presented in Fig. 6.

The students are managed to find confidential information that they had searched with previous methods. They get the sensitive information (flag) in the form of a password. Furthermore, this password is utilized to submit-flag to acquire the points.


sniffing_DdM3w2D.pcapng [Wireshark 1.12.7 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

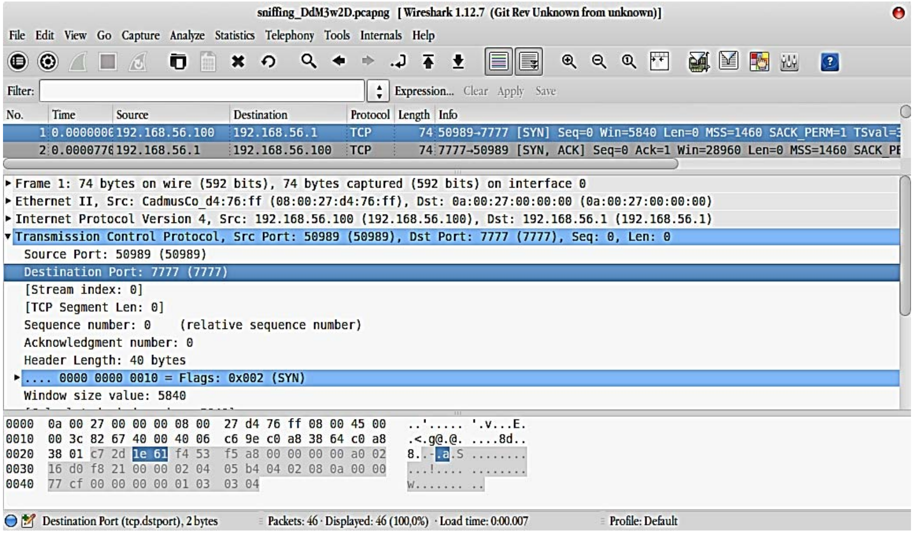
```
No.  Time      Source           Destination      Protocol Length Info
   1 0.0000000 192.168.56.100   192.168.56.1     TCP          74 50989→7777 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=3
   2 0.0000776 192.168.56.1     192.168.56.100   TCP          74 7777→50989 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PE
   3 0.0012586 192.168.56.100   192.168.56.1     TCP          66 50989→7777 [ACK] Seq=1 Ack=1 Win=5840 Len=0 TSval=30671 TSecr=1204
   4 5.0092836 0a:00:27:00:00:0 CadmusCo_d4:76:f ARP          42 Who has 192.168.56.100?  Tell 192.168.56.1
   5 5.0100046 CadmusCo_d4:76:f 0a:00:27:00:00:0 ARP          42 192.168.56.100 is at 08:00:27:d4:76:ff
   6 25.263809 192.168.56.100   192.168.56.255   BROWSEI     286 Host Announcement METASPLOITABLE, Workstation, Server, Print Queue
   7 25.263834 192.168.56.100   192.168.56.255   BROWSEI     286 Host Announcement METASPLOITABLE, Workstation, Server, Print Queue
   8 55.306101 192.168.56.1     192.168.56.255   BROWSEI     271 Local Master Announcement LUPIT, Workstation, Server, Print Queue
   9 55.306176 192.168.56.1     192.168.56.255   BROWSEI     248 Domain/Workgroup Announcement WORKGROUP, NT Workstation, Domain Er
  10 67.547932 192.168.56.100   192.168.56.1     TCP          84 50989→7777 [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=18 TSval=37425 TSec

▸ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: CadmusCo_d4:76:ff (08:00:27:d4:76:ff), Dst: 0a:00:27:00:00:00 (0a:00:27:00:00:00)
▸ Internet Protocol Version 4, Src: 192.168.56.100 (192.168.56.100), Dst: 192.168.56.1 (192.168.56.1)
▸ Transmission Control Protocol, Src Port: 50989 (50989), Dst Port: 7777 (7777), Seq: 0, Len: 0

0000  0a 00 27 00 00 00 08 00  27 d4 76 ff 08 00 45 00   ..'..... '.v...E.
0010  00 3c 82 67 40 00 40 06  c6 9e c0 a8 38 64 c0 a8   .<.g@.@. ....8d..
0020  38 01 c7 2d 1e 61 f4 53  f5 a8 00 00 00 00 a0 02   8..-.a.S ........
0030  16 d0 f8 21 00 00 02 04  05 b4 04 02 08 0a 00 00   ...!.... ........
0040  77 cf 00 00 00 00 01 03  03 04                     w....... ..
```

File: "/tmp/mozilla_root0/sniffing_DdM3w2… Packets: 46 · Displayed: 46 (100,0%) · Load time: 0:00.007 Profile: Default

Fig. 3. Wireshark Interface and the content downloaded file.


sniffing_DdM3w2D.pcapng [Wireshark 1.12.7 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

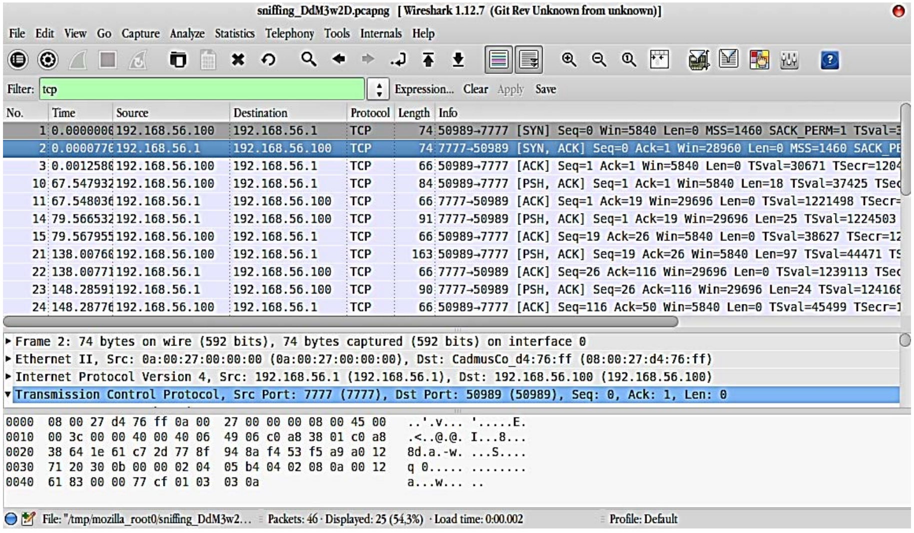
```
No.  Time      Source           Destination      Protocol Length Info
   1 0.0000000 192.168.56.100   192.168.56.1     TCP          74 50989→7777 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=3
   2 0.0000776 192.168.56.1     192.168.56.100   TCP          74 7777→50989 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PE

▸ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: CadmusCo_d4:76:ff (08:00:27:d4:76:ff), Dst: 0a:00:27:00:00:00 (0a:00:27:00:00:00)
▸ Internet Protocol Version 4, Src: 192.168.56.100 (192.168.56.100), Dst: 192.168.56.1 (192.168.56.1)
▾ Transmission Control Protocol, Src Port: 50989 (50989), Dst Port: 7777 (7777), Seq: 0, Len: 0
    Source Port: 50989 (50989)
    Destination Port: 7777 (7777)
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    Acknowledgment number: 0
    Header Length: 40 bytes
  ▸ .... 0000 0000 0010 = Flags: 0x002 (SYN)
    Window size value: 5840

0000  0a 00 27 00 00 00 08 00  27 d4 76 ff 08 00 45 00   ..'..... '.v...E.
0010  00 3c 82 67 40 00 40 06  c6 9e c0 a8 38 64 c0 a8   .<.g@.@. ....8d..
0020  38 01 c7 2d 1e 61 f4 53  f5 a8 00 00 00 00 a0 02   8..-.a.S ........
0030  16 d0 f8 21 00 00 02 04  05 b4 04 02 08 0a 00 00   ...!.... ........
0040  77 cf 00 00 00 00 01 03  03 04                     w....... ..
```

Destination Port (tcp.dstport), 2 bytes Packets: 46 · Displayed: 46 (100,0%) · Load time: 0:00.007 Profile: Default

Fig. 4. TCP/IP layer inspection on wireshark.


sniffing_DdM3w2D.pcapng [Wireshark 1.12.7 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save

Filter: tcp

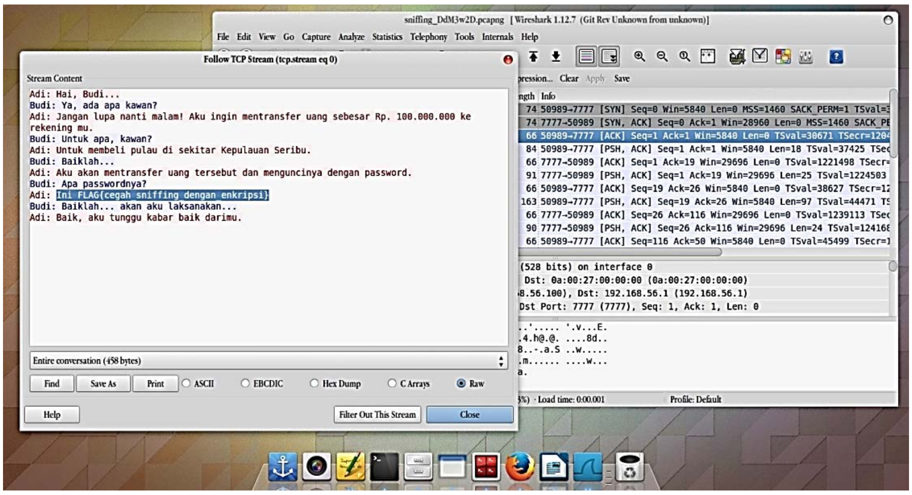
```
No.  Time      Source           Destination      Protocol Length Info
   1 0.0000000 192.168.56.100   192.168.56.1     TCP          74 50989→7777 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=3
   2 0.0000776 192.168.56.1     192.168.56.100   TCP          74 7777→50989 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PE
   3 0.0012586 192.168.56.100   192.168.56.1     TCP          66 50989→7777 [ACK] Seq=1 Ack=1 Win=5840 Len=0 TSval=30671 TSecr=1204
  10 67.547932 192.168.56.100   192.168.56.1     TCP          84 50989→7777 [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=18 TSval=37425 TSec
  11 67.548036 192.168.56.1     192.168.56.100   TCP          66 7777→50989 [ACK] Seq=1 Ack=19 Win=29696 Len=0 TSval=1221498 TSecr=
  14 79.566532 192.168.56.1     192.168.56.100   TCP          91 7777→50989 [PSH, ACK] Seq=1 Ack=19 Win=29696 Len=25 TSval=1224503
  15 79.567955 192.168.56.100   192.168.56.1     TCP          66 50989→7777 [ACK] Seq=19 Ack=26 Win=5840 Len=0 TSval=38627 TSecr=12
  21 138.00766 192.168.56.100   192.168.56.1     TCP         163 50989→7777 [PSH, ACK] Seq=19 Ack=26 Win=5840 Len=97 TSval=44471 TS
  22 138.00771 192.168.56.1     192.168.56.100   TCP          66 7777→50989 [ACK] Seq=26 Ack=116 Win=29696 Len=0 TSval=1239113 TSec
  23 148.28591 192.168.56.1     192.168.56.100   TCP          90 7777→50989 [PSH, ACK] Seq=26 Ack=116 Win=29696 Len=24 TSval=124168
  24 148.28776 192.168.56.100   192.168.56.1     TCP          66 50989→7777 [ACK] Seq=116 Ack=50 Win=5840 Len=0 TSval=45499 TSecr=1

▸ Frame 2: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: 0a:00:27:00:00:00 (0a:00:27:00:00:00), Dst: CadmusCo_d4:76:ff (08:00:27:d4:76:ff)
▸ Internet Protocol Version 4, Src: 192.168.56.1 (192.168.56.1), Dst: 192.168.56.100 (192.168.56.100)
▾ Transmission Control Protocol, Src Port: 7777 (7777), Dst Port: 50989 (50989), Seq: 0, Ack: 1, Len: 0

0000  08 00 27 d4 76 ff 0a 00  27 00 00 00 08 00 45 00   ..'.v... '.....E.
0010  00 3c 00 00 40 00 40 06  49 06 c0 a8 38 01 c0 a8   .<..@.@. I...8...
0020  38 64 1e 61 c7 2d 77 8f  94 8a f4 53 f5 a9 a0 12   8d.a.-w. ...S....
0030  71 20 30 0b 00 00 02 04  05 b4 04 02 08 0a 00 12   q 0..... ........
0040  61 83 00 00 77 cf 01 03  03 0a                     a...w... ..
```

File: "/tmp/mozilla_root0/sniffing_DdM3w2… Packets: 46 · Displayed: 25 (54,3%) · Load time: 0:00.002 Profile: Default

Fig. 5. Filtering TCP packet on wireshark.



sniffing_DdM3w2D.pcapng [Wireshark 1.12.7 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Follow TCP Stream (tcp.stream eq 0)
Stream Content
Entire conversation (458 bytes)
Find
Save As
Print
○ ASCII
○ EBCDIC
○ Hex Dump
○ C Arrays
● Raw
Help
Filter Out This Stream
Close
pression... Clear Apply Save
ngth Info
3%) · Load time: 0:00.001
Profile: Default

```
Adi: Hai, Budi...
Budi: Ya, ada apa kawan?
Adi: Jangan lupa nanti malam! Aku ingin mentransfer uang sebesar Rp. 100.000.000 ke
rekening mu.
Budi: Untuk apa, kawan?
Adi: Untuk membeli pulau di sekitar Kepulauan Seribu.
Budi: Baiklah...
Adi: Aku akan mentransfer uang tersebut dan menguncinya dengan password.
Budi: Apa passwordnya?
Adi: Ini FLAG{cegah sniffing dengan enkripsi}
Budi: Baiklah... akan aku laksanakan...
Adi: Baik, aku tunggu kabar baik darimu.
```

```
 74 50989→7777 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=3
 74 7777→50989 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PE
 66 50989→7777 [ACK] Seq=1 Ack=1 Win=5840 Len=0 TSval=30671 TSecr=120
 84 50989→7777 [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=18 TSval=37425 TSe
 66 7777→50989 [ACK] Seq=1 Ack=19 Win=29696 Len=0 TSval=1221498 TSecr=
 91 7777→50989 [PSH, ACK] Seq=1 Ack=19 Win=29696 Len=25 TSval=1224503
 66 50989→7777 [ACK] Seq=19 Ack=26 Win=5840 Len=0 TSval=38627 TSecr=12
163 50989→7777 [PSH, ACK] Seq=19 Ack=26 Win=5840 Len=97 TSval=44471 TS
 66 7777→50989 [ACK] Seq=26 Ack=116 Win=29696 Len=0 TSval=1239113 TSec
 90 7777→50989 [PSH, ACK] Seq=26 Ack=116 Win=29696 Len=24 TSval=12416
 66 50989→7777 [ACK] Seq=116 Ack=50 Win=5840 Len=0 TSval=45499 TSecr=1
```

```
(528 bits) on interface 0
 Dst: 0a:00:27:00:00:00 (0a:00:27:00:00:00)
8.56.100), Dst: 192.168.56.1 (192.168.56.1)
Dst Port: 7777 (7777), Seq: 1, Ack: 1, Len: 0
```

```
..'..... '.v...E.
.4.h@.@. ....8d..
8..-.a.S ..w.....
.m...... ....w...
a.
```

Fig. 6. The results of confidential information successfully discovered.

III. Capture the Flag and Boyer-Moore Algorithm

The Boyer-Moore algorithm is the most efficient string-matching algorithm that has been widely applied in text editor applications [7]. A simple version of the algorithm is often implemented in text editor for search and substitute commands.

The algorithm effectively scans the character pattern from right to left. In case of incompatibility, the Boyer-Moore algorithm uses the pre-computed function to move the window to the right. These two shift functions are called *good-suffix shift* (Fig. 7) and *bad-character shift* (Fig. 8).

Assume a mismatch occurs between the characters x [i] = a of pattern and character y [i + j] = b, during the experiment of j position. Then, x [i + 1 ... m-1] = y [i + j + 1 ... j + m-1] = u and x [i]! = Y [i + j]. Good-suffix shift consists of segments y [i + j + 1 ... j + m-1] = x [i + 1 ... m-1] with the rightmost segment in x preceded by a different character from x [i].

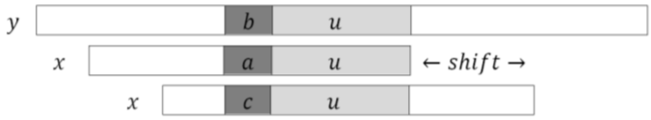


Fig. 7. Good-suffix shift, *u* occurs repeatedly.

If there is a segment, shift consists of the longest suffix v of y [i + j + 1 ... j + m-1] with a prefix-match of x

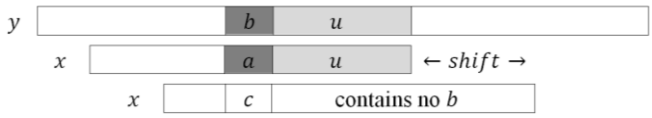


Fig. 8. Bad-suffix shift, *a* occurs in *x*.

Bad-character shift consists of text characters arranged by y [i + j] with the rightmost segments on x [0 ... m-2].

Note: bad-character shift may be negative; therefore to shift the window, the Boyer-Moore algorithm applies the maximum value between good-suffix shift and bad-character shift. Formally, these two shift functions are defined as follows:

Cs (i,s): For every k that meets the conditions of i <k <m, s> = k or x [k-s] = x [k] and,

Co (i, s): If s <i then x [i-s]! = x [i].

Afterwards, for 0 <= i <m: bmGs [i + 1] = min {s> 0: Cs (i, s) and Co (i, s)} and we can define bmGs [0] as the length of period x.

In CTF network security learning media, the Boyer-Moore algorithm is applied to automatically assess Write-up, in which the role is for searching a match between the teacher's predefined keyword on the problem with the problem-solving report that contains of students' knowledge of the problem.

The use of the Boyer-Moore algorithm takes the assumption that the longer the pattern is searched; the search time performed by the Boyer-Moore algorithm gets shorter. This is because if the pattern is long then the shift that can be done is also greater. The best case for Boyer Moore's algorithm is when the first character on the comparable text does not exist in the pattern. If this happens frequently then the search time will be shorter [8] [9] [10].

The result of the matching test will be converted as players earn point and subsequently are accumulated to the team points on the Scoreboard. The Boyer-Moore algorithm works based on keywords that have been defined by the teachers, for example, on the Sniffing problem, the key word is sniffing, IP, and encryption. The Illustration of string matching in Write-up using the Boyer-Moore algorithm is presented in Fig. 9.

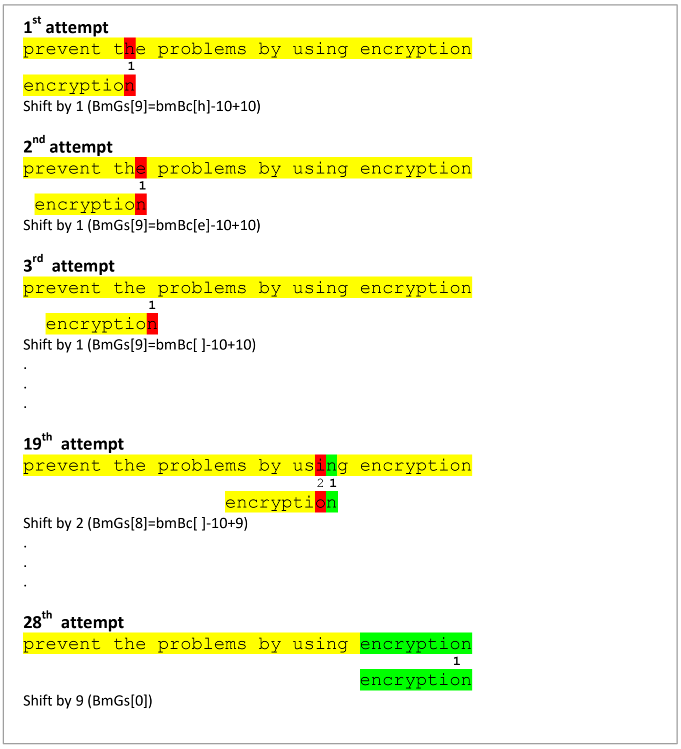

1st attempt
prevent the problems by using encryption
1
encryption
Shift by 1 (BmGs[9]=bmBc[h]-10+10)
2nd attempt
prevent the problems by using encryption
1
encryption
Shift by 1 (BmGs[9]=bmBc[e]-10+10)
3rd attempt
prevent the problems by using encryption
1
encryption
Shift by 1 (BmGs[9]=bmBc[ ]-10+10)
.
.
.
19th attempt
prevent the problems by using encryption
2 1
encryption
Shift by 2 (BmGs[8]=bmBc[ ]-10+9)
.
.
.
28th attempt
prevent the problems by using encryption
1
encryption
Shift by 9 (BmGs[0])

Fig. 9. Boyer-Moore algorithm of string matching.

In this simulation, the Boyer-Moore algorithm runs carefully and determines the position by the precise precision. The Boyer-Moore algorithm implemented in CTF network security learning media is built with the Python programming language, and Django framework. The source code of the Boyer-Moore algorithm by using the Python programming language is expressed in Fig. 10.

```
# Boyer-moore searching phase def
bmsrch(hystck, needl):
    goodsfx = gensfxshft(needl) badchr =
    genbadchrshft(needl) i = 0
    while i < len(hystck)-len(needl)+1: j =
        len(needl)
        while j > 0 and needl[j-1] == hystck[i+j-1]:
                j -= 1
            if j > 0:
                badchrshft =
badchr.get(hystck[i+j-1], len(needl))
                goodsfxshft =
goodsfx[len(needl)-j]
                if badchrshft > goodsfxshft: i +=
                    badchrshft
                  else:
                    i += goodsfxshft
            else:
                return i
    return -1
 if __name__ == '__main__':
     blob = "Cegah masalah Sniffing dengan enkripsi "
    keyword = "enkripsi"
    print bmsrch(blob, keyword)
```

Fig. 10. Algorithm Boyer-Moore Source Code.

IV. Experiments and Discussion

The experiments of developed media were executed by involving 34 students and five experts in the field of network and the development of learning support technology. The instrument used in multimedia retrieval refers to the Learning Object Review Instrument (LORI) version 1.5 [11]. Based on the results of expert assessment, which includes aspects of Design Presentation, Interaction Usability, Accessibility, Reusability, Standards Compliance, Content Quality, Learning Goal Alignment, Feedback and Adaptation, and Motivation, the developed media are considered appropriate to apply and to be used as learning reference with overall average score of 4.5. The details of the media assessment results are presented in Table 1.

Through teacher instruction, the students enable to engage in CTF learning scenarios designed within the media and solve the problems of threats and attacks on network security as well as apply network security practices using the sniffer, which is Wireshark. At first, they need time to habituate with the media interaction; nevertheless they tend to give positive responses to the developed media. They are eager and motivated to learn, try and understand the subjects of network security. Some feel challenged to be able to compete with other teams. Cognitively, students' comprehension skills increased by 0.61 with moderate criteria based on the formulations offered by Hake in the measurement of enhancement capabilities [12]. The improvement that occurred at the moderate level was due to changes in learning patterns from previous traditional approaches that were dominated by teacher activity into student -centered learning- and the teacher as the facilitator and mediator.

TABLE I. CTF Learning Object Review

Aspect	Ideal Score	Expert Score					Avr.	LORI Score
		1	2	3	4	5		
Content Quality	20	18	17	19	19	18	18.2	4.55
Learning Goal Alignment	20	17	18	19	17	18	17.8	4.45
Feedback and Adaptation	5	5	4	4	5	4	4.4	4.4
Motivation	5	5	5	4	5	5	4.4	4.4
Presentation Design	5	5	4	5	4	4	4.8	4.8
Interaction Usability	15	13	15	14	14	15	14.2	4.73
Accessibility	10	9	9	9	9	8	8.8	4.4
Reusability	5	4	5	5	4	5	4.6	4.6
Standards Compliance	5	4	5	4	4	4	4.2	4.2

V. Conclutions and Suggestion for Future Work

The main contribution of this study lies in the 'simplification' of complex material presentation forms into simpler, easier to understand and self-engagement impacts. On the other hand, this study also provides opportunities for researchers and teachers to develop network security capabilities resulting in increased student's interest in network security and opportunities for students to engage in digital forensic activities.

As future work, CTF-related developments need to consider the analysis of related knowledge, such as technology knowledge, pedagogy knowledge, content knowledge and slices among the three, commonly known as Technological Pedagogical Content Knowledge (TPACK) [13-15].

ACKNOWLEDGMENT

Grateful to the Department of Computer Science Education, Universitas Pendidikan Indonesia who have given support to the writing of this manuscript

REFERENCES

[1] G. Vigna, "Teaching Hands-on Network Security : Testbed and Live Exercises," *Journal of Information Welfare,* vol. 2, no. 3, pp. 8-24, 2003.

[2] J. Mirkovic, M. Ryan, J. Hickey, K. Sklower, P. Reiher, P. A. H. Peterson, B. H. Kang, M. C. Chuah, D. Massey and G. Ragusa, "Teaching Security With Network Testbeds," in *ACM Sigcomm Educational Workshop*, 2011.

[3] CTF TIME, "CTF? WTF?," Transdata, 2012. [Online]. Available: https://ctftime.org/ctf-wtf/. [Accessed 19 April 2016].

[4] M.N.M. Khambari, M. F. I. Othman, M. R. Motsidi and M. F. Abdollah, "A Novel Approach on Teaching Network Security for ICT Courses," in *2009 International Conference on Engineering Education (ICEED)* , Kuala Lumpur - Malaysia, 2009.

[5] A. Knowles, "Behind the Scenes at a Capture the Flag (CTF) Competition," Security Intelligence, 8 December 2016. [Online]. Available: https://securityintelligence.com/behind-the-scenes-at-a-capture-the-flag-ctf-competition/. [Accessed 12 May 2017].

[6] G. Vigna, "Teaching Network Security through Live Exercises," in *Third Annual World Conference on Information Security Education (WISE3)*, Monterey, California, USA, 2003.

[7] C. Charras and T. Lecroq, Handbook of Exact String Matching Algorithms, King's College London Publications, 2004.

[8] P. Jokinen, J. Tarhio and E. Ukkonen, "A Comparison of Approximate String Matching Algorithms," *SOFTWARE—PRACTICE AND EXPERIENCE,* vol. 1, no. 1, pp. 1-19, January 1988.

[9] L. Salmela and J. Tarhio, "Approximate String Matching with Reduced Alphabet," in *Algorithms and Applications*, Berlin, Springer, Heidelberg, 2010, pp. 210-220.

[10] P. Jain and S. Pandey, "Comparative Study on Text Pattern Matching for Heterogeneous System," *International Journal of Computer Science & Engineering Technology (IJCSET),* vol. 3, no. 11, pp. 537-543, 2012.

[11] J. Nesbit, K. Belfer and T. Leacock, "Learning Object Review Instrument (LORI) version 1.5 - User Manual," http://www.elera.net, 2004.

[12] R.R. Hake, "Departement of Physics Indiana University," 19 June 1999. [Online]. Available: http://www.physics.indiana.edu/~sdi/AnalyzingChange-Gain.pdf. [Accessed 10 March 2011].

[13] M. Koehler and P. Mishra, "Introducing TPCK," in *The handbook of technological pedagogical content knowledge (TPCK) for educators*, New York, Routledge, 2008, pp. 3-29.

[14] M.C. Herring, P. Mishra and M. J. Koehler, Handbook of Technological Pedagogical Content Knowledge (TPCK) for Educators, Routledge: Taylor & Francis Group for the American Association of Colleges for Teacher Education (AACTE), 2014.

[15] M.J. Koehler and P. Mishra, "What Happens when Teachers Design Educational Technology? The Development of Technological Pedagogical Content Knowledge," *Journal Educational Computing Research,* vol. 32, no. 2, pp. 131-152, 2005.