Quantitative Usability Evaluation

COMP 3008 Project 2

Due: April 6th, 2018

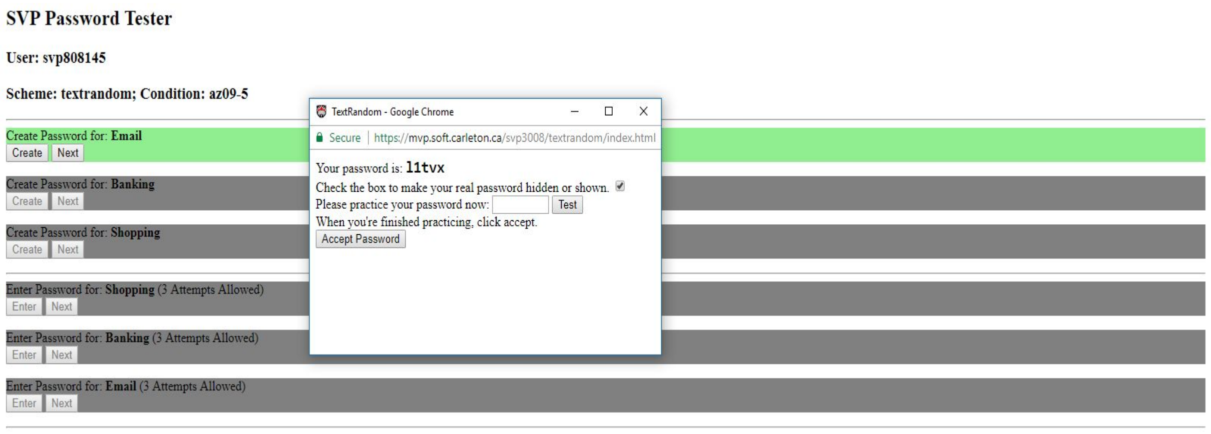
<u>Group Members</u>

Thuong Mai 100885938
Syed Arsal Abbas 100835968
Muhammad Junaid Farooqui 100912649

Table of Content

1. Sample Data and Descriptive Statistics

For this project, we considered two password schemes. The first password scheme is Text21. This is a text-based scheme and it prompts the user with a randomly generated password of 4 random letters chosen from alphabets a - z, and a number chosen from 0 – 9 and displays it. Following is a set of screenshots (Figure 1 & Figure 2) taken to demonstrate how this scheme displays a randomly generated password for a user:


SVP Password Tester

User: svp808145

Scheme: textrandom; Condition: az09-5

Create Password for: Email
Create Next

Create Password for: Banking
Create Next

Create Password for: Shopping
Create Next

Enter Password for: Shopping (3 Attempts Allowed)
Enter Next

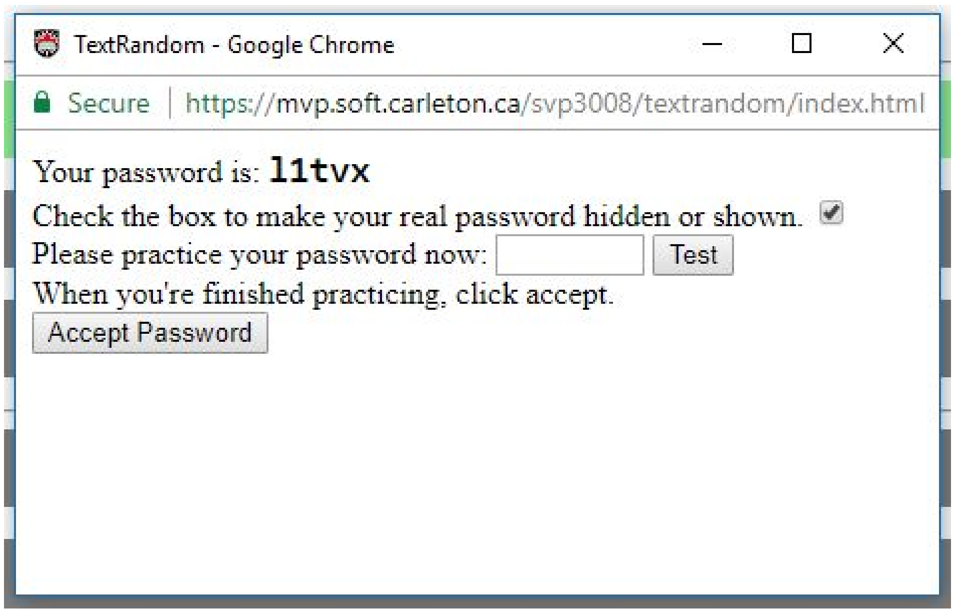
Enter Password for: Banking (3 Attempts Allowed)
Enter Next

Enter Password for: Email (3 Attempts Allowed)
Enter Next

TextRandom - Google Chrome
Secure | https://mvp.soft.carleton.ca/svp3008/textrandom/index.html

Your password is: 11tvx
Check the box to make your real password hidden or shown.
Please practice your password now: Test
When you're finished practicing, click accept.
Accept Password

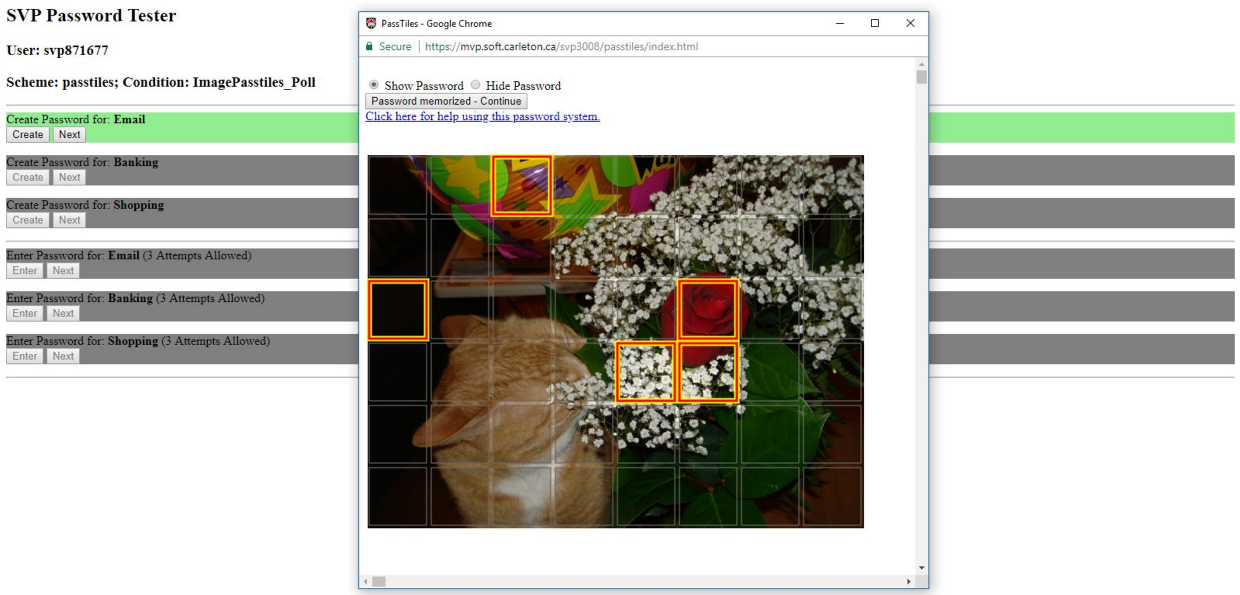
Figure 1 - Text21 scheme


TextRandom - Google Chrome
Secure | https://mvp.soft.carleton.ca/svp3008/textrandom/index.html

Your password is: 11tvx
Check the box to make your real password hidden or shown.
Please practice your password now: Test
When you're finished practicing, click accept.
Accept Password

Figure 2 - Text21 scheme

The second password scheme is Imagept21. It displays an 8x6 grid with five randomly chosen tiles in the grid as the password is generated for the user. The chosen tiles are highlighted with a colour to facilitate the user to memorize the password easily. A display picture is added as a background on the 48-tile grid, so it looks more like a puzzle. The following set of screenshots (Figure 3 & Figure 4) show how this scheme is used to display a randomly generated password for the user:



SVP Password Tester

User: svp871677

Scheme: passtiles; Condition: ImagePasstiles_Poll

Create Password for: **Email**

Create Password for: **Banking**

Create Password for: **Shopping**

Enter Password for: **Email** (3 Attempts Allowed)

Enter Password for: **Banking** (3 Attempts Allowed)

Enter Password for: **Shopping** (3 Attempts Allowed)

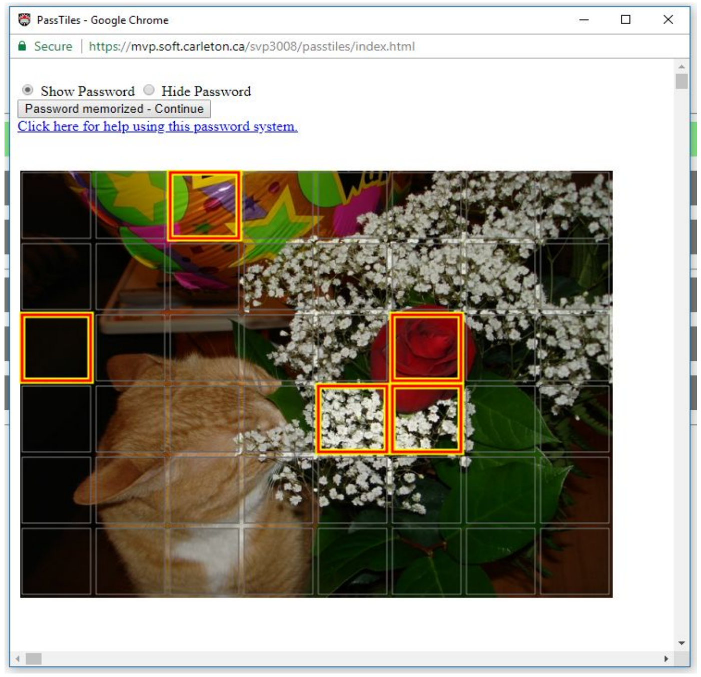
PassTiles - Google Chrome

Secure | https://mvp.soft.carleton.ca/svp3008/passtiles/index.html

● Show Password ○ Hide Password

Click here for help using this password system.

Figure 3 - Imagept21 scheme



PassTiles - Google Chrome

Secure | https://mvp.soft.carleton.ca/svp3008/passtiles/index.html

● Show Password ○ Hide Password

Click here for help using this password system.

Figure 4 - Imagept21 scheme

1.1 Advantages and Disadvantages

Both above-mentioned password schemes have their pros and cons. In this this section of the report we will take a look at the advantages and disadvantages of each scheme respectively. *Text21*, while not actually used in practice, is the most commonly recognized password scheme in today's world (i.e. most passwords that people use are alphanumeric and are not fixed to a random permutation of 4 lowercase characters and a number). Having said that, *Text21* is guaranteed a 2^{25} password space. This password space translates roughly into $2^{25.4373142063}$ (to be exact) bits of entropy, however, for the average user, a randomly generated password of characters and numbers is most certainly not that easy to remember. Despite *Text21* only using 5 letters for the password itself, the characters are not guaranteed to have a coherent pattern.

Imagept21 strives to solve the shortcomings of *Text21* by replacing the text-based password approach with a pattern-based approach. The premise of *Imagept21* uses human being's natural pattern-recognition skills to reinforce a password by selecting five random tiles in an 8x6 grid. Naturally, the password space of *Imagept21* is translating approximately to $2^{27.9248125036}$. This scheme, as mentioned above, uses a picture as a background on a 48-tile grid and user is prompted to select randomly generated 5 tiles for a password, which adds a benefit of allowing a user to make and memorize patterns rather more quickly and easily.

Out of the above mentioned two schemes, people who utilize pattern-based recognition more than straight memorization would pick *Imagept21*. On the other hand, people who prefer a more compact or a text-based password scheme would choose *Text21.*

1.2 Documentation

We were given the logged data to begin with, and to process this logged data, we used R programming language. The data given in the files Text21.csv and Imagept21.csv was organized as timestamps of each event. Each row in the files had the following fields: time, user, site, scheme, mode, event and data; and each file was ordered by time. Team 48 decided to keep the data sorted by time. This way we could keep track of when a login attempt was made and when that login attempt was finished. Our team also decided that the resulting csv files should contain the following information: userid, site, password scheme, event (success or failure), and the time taken for each login.

In order to get each login attempt we iterated through the data with a for loop and kept track of each row where the event would start. Then we looked into the next row where the event was either a “success” or a “failure”. In an attempt to get the "success" and "failure" events, we ignored every other events like “create”, “register” and “verifytest” etc. as they had nothing to do with a login attempt. Then we achieved the time difference between these two events and added a new row to a data frame with all the required/important information. We had a few cases where the user would start a login and then not complete it and start again. To fix those issues we had another IF statement inside a for loop that made sure that there was a “start” event before a “success” or a “failure” event. This continued for each successful or failed login attempt. The for loop that was used to iterate through the whole file was created inside a function called “fun”, this way we could easily repeat the process for each log data file. For instance, fun(text21) would get the results for the Text21.csv file.

There were two resulting data files produced from this process: Text21Result.csv and Imagept21Result.csv, that can be found in the provided folder “R”. The source code, “function.R”, can be found in the same folder as well. The source code includes the code that uses the resulting data to produce the tables and graphs in the following comparison section 1.3.

For better readability, we have attached the R source code in this document:

```
#reading the csv files
text21 = read.csv("Text21.csv", row.names = NULL)
image21 = read.csv("Imagept21.csv", row.names = NULL)

#function(inputFile)
fun = function(inputFile) {
       start = FALSE
       df = data.frame(NA, NA, NA, NA, NA)
       names(df) = c("user", "site", "scheme", "event", "time")

       for (i in 1:dim(inputFile)[1]) {
         if (inputFile$event[i] == "start") {
           #storing the time in t1 when an even "start" occurs
           t1 = inputFile$time[i]
           user = inputFile$user[i]
           start = TRUE
         }

       #catching the events "success" and "failure"
       if (inputFile$event[i] == "success" || inputFile$event[i] == "failure") {

           #catching the mode "login" IFF the events "success" and "failure" has occurred
           if (start == TRUE &&
              inputFile$mode[i] == "login" &&
              inputFile$user[i] == user) {

              #storing the time in t1 when an even "start" occurs
              t2 = inputFile$time[i]

              #storing the time difference between t2 and t1 and string the value in t3
              with the units "secs"
```

```
               t3 = difftime(
                  strptime(t2, format = "%Y-%m-%d %H:%M:%S", tz = ""),
                  strptime(t1, format = "%Y-%m-%d %H:%M:%S", tz = ""),
                  units = "secs"
               )

       #Checking and Getting rid of outliers which are > 2min and adding the value t3 to the
newly created dataframe

               if (t3 < 120) {
                 row = c(
                    toString(inputFile$user[i]),
                    toString(inputFile$site[i]),
                    toString(inputFile$scheme[i]),
                    toString(inputFile$event[i]),
                    t3
               )

                  df = rbind(row, df)
                  start = FALSE
                }
              }
            }
          }

df = df[-c(dim(df)), ] #removing the last row (NA,NA...)
df$time = as.numeric(as.character(df$time)) #setting the time col into int

user = data.frame(table(user = df$user))
userEvent = data.frame(table(user = df$user, event = df$event))

#Total Number of Logins
#Mean, Median and Standard Deviation for number of total logins
user$Freq = as.numeric(as.character(user$Freq))
meanTotal = mean(user$Freq)
medianTotal = median(user$Freq)
sdTotal = sd(user$Freq)

#Mean, Median and Standard Deviation for number of successful logins
numSuccess = userEvent[-grep("failure", userEvent$event), ]
numSuccess$event = NULL
numSuccess$Freq = as.numeric(as.character(numSuccess$Freq))
meanSuccess = mean(numSuccess$Freq)
medianSuccess = median(numSuccess$Freq)
sdSuccess = sd(numSuccess$Freq)

#Mean, Median and Standard Deviation for number of failed logins
numFailure = userEvent[-grep("success", userEvent$event), ]
numFailure$event = NULL
numFailure$Freq = as.numeric(as.character(numFailure$Freq))
meanFailure = mean(numFailure$Freq)
medianFailure = median(numFailure$Freq)
sdFailure = sd(numFailure$Freq)

#Mean, Median and Standard Deviation (in short: MMS)
MMS = data.frame(NA, NA, NA, NA)
```

```
Total = c("Total", meanTotal, medianTotal, sdTotal)
MMS = rbind(Total, MMS)
Failure = c("Failure", meanFailure, medianFailure, sdFailure)
MMS = rbind(Failure, MMS)
Success = c("Success ", meanSuccess, medianSuccess, sdSuccess)
MMS = rbind(Success, MMS)
MMS = MMS[-c(dim(MMS)), ] #removing the last row (NA,NA...)
names(MMS) = c("Logins", "Mean", "Median", "Standard Deviation")
View(MMS) #Mean, Median and Standard Deviation Total Login tables

#Number of Total logins table
logins = merge(numSuccess, numFailure, by = "user")
logins = merge(logins, user, by = "user")
names(logins) = c("users", "successful logins", "failed logins", "total logins")
View(logins) #Total Login tables for users


#Login Time
#Mean, Median and Standard Deviation for successful login times
success = df[-grep("failure", df$event), ]
successTimes = data.frame(sapply(split(success$time, success$user), mean),
                          sapply(split(success$time, success$user), median),
                          sapply(split(success$time, success$user), sd))

names(successTimes) = c("mean", "median", "standard deviation")
View(successTimes) #Success times tables

#Mean, Median and Standard Deviation for failed login times
failure = df[-grep("success", df$event), ]

failureTimes = data.frame(sapply(split(failure$time, failure$user), mean),
                          sapply(split(failure$time, failure$user), median),
                          sapply(split(failure$time, failure$user), sd))

names(failureTimes) = c("mean", "median", "standard deviation")
View(failureTimes) #Failure times table

#Histogram for number of logins
hist(user$Freq,
     main = "Total Number of logins",
     xlab = "Logins",
     border = "black",
     col = "khaki3")

#Histogram for Number of successful logins
hist(numSuccess$Freq,
     main = "Number of successful logins",
     xlab = "Successful logins",
     border = "black",
     col = "green")


#Histogram for Number of failed logins
hist(numFailure$Freq,
     main = "Number of failed logins",
     xlab = "Failed logins",
     border = "black",
     col = "red")

#Histogram for Successful login times
hist(successTimes$mean,
     main = "Successful login times",
     xlab = "Success time(s)",
     border = "black",
     col = "green")

#Histogram for Failed login times
hist(failureTimes$mean,
     main = "Failed login times",
     xlab = "Failure time(s)",
     border = "black",
     col = "red")

#BoxPlot for Successful login times
boxplot(successTimes$mean,
     main = "Successful login times",
     border = "black",
     col = "green")

#BoxPlot for Failed login times
boxplot(failureTimes$mean,
     main = "Failed login times",
     border = "black",
     col = "red")

write.csv(df, file = "Results.csv")
}

       #~~~~~~~~~~~~You need to uncomment these function calls one at a time, to produce the
desired results.~~~~~~~~~~~~#

       #fun(text21)
       #fun(image21)
```

1.3 Scheme usability comparison

The source code, “function.R”, can be found in the seperate “R” folder provided. As shown in the above section 1.2, the source code includes code that gets the resulting data from the log data and code that uses the resulting data to produce the tables and graphs. Lines 58 to 167 is for producing tables and graphs of the resulting data. They were both included in the same function to simplify the work. Below you will find the tables and graphs of the resulting data for each password scheme.

1.3.1 Text21 Data

User	Successful Logins	Failed Logins	Total Logins	Success %
ast103	16	1	17	94.12
ast104	16	10	26	61.54
ast105	12	3	15	80
ast107	15	6	21	71.43
ast108	19	0	19	100
ast111	11	8	19	57.89
ast112	15	0	15	100
ast114	15	7	22	68.18
ast115	15	0	15	100
ast116	15	0	15	100
ast118	15	1	16	93.75
ast125	15	3	18	83.33
ast131	15	6	21	71.43
ast133	9	0	9	100
ast134	15	0	15	100
ast135	3	1	4	75
ast136	15	0	15	100
ast138	15	0	15	100

Table 1 - Number of logins per user

Logins	**Mean**	**Median**	**Standard Deviation**
Success	13.94444	15	3.455127855
Failure	2.555556	1	3.329409455
Total	16.5	15.5	4.877921447

Table 2 - Mean, median and standard deviation for logins

User	**Mean**	**Median**	**Standard Deviation**
ast103	12.1875	6	13.55590277
ast104	11.9375	11	3.957587649
ast105	7.083333333	7	2.108783938
ast107	5.133333333	5	1.807392228
ast108	4.210526316	3	2.699360981
ast111	7.818181818	8	1.990888335
ast112	7.733333333	7	2.086236073
ast114	11.13333333	10	4.470005859
ast115	11.2	9	6.014268748
ast116	5.933333333	6	1.486446706
ast118	7.8	7	1.740279124
ast125	6.4	5	3.459975227
ast131	12.06666667	8	7.391758603
ast133	14.55555556	12	5.703312877
ast134	7.533333333	5	3.700707783
ast135	7.333333333	7	1.527525232
ast136	10.33333333	9	4.064948896
ast138	15.8	15	5.505841054

Table 3 - Successful login times per user

<u>User</u>	<u>Mean</u>	<u>Median</u>	<u>Standard Deviation</u>
ast103	18	18	NA
ast104	14	13	5.120764
ast105	8.333333333	7	5.131601
ast107	6.5	5	4.230839
ast111	9.125	7	4.911721
ast114	9.857142857	10	2.91139
ast118	9	9	NA
ast125	6	6	2
ast131	22.5	10	20.94517
ast135	5	5	NA

Table 4 - Failed login times per user

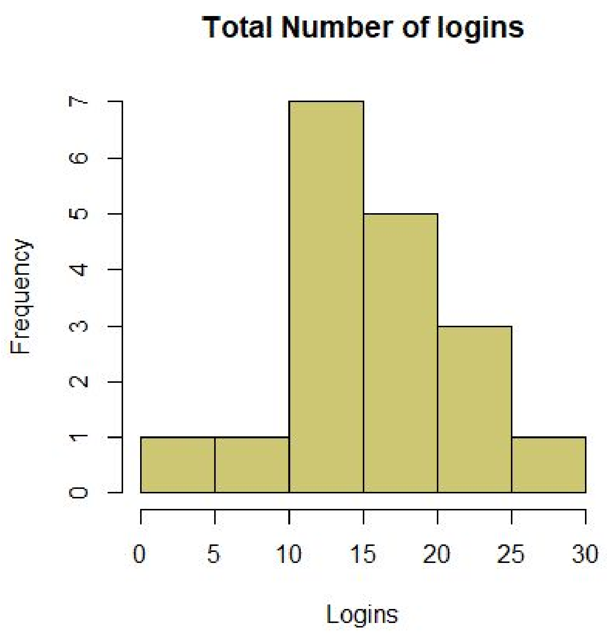


Figure 5 - Histogram for total number of logins (Text21 scheme)

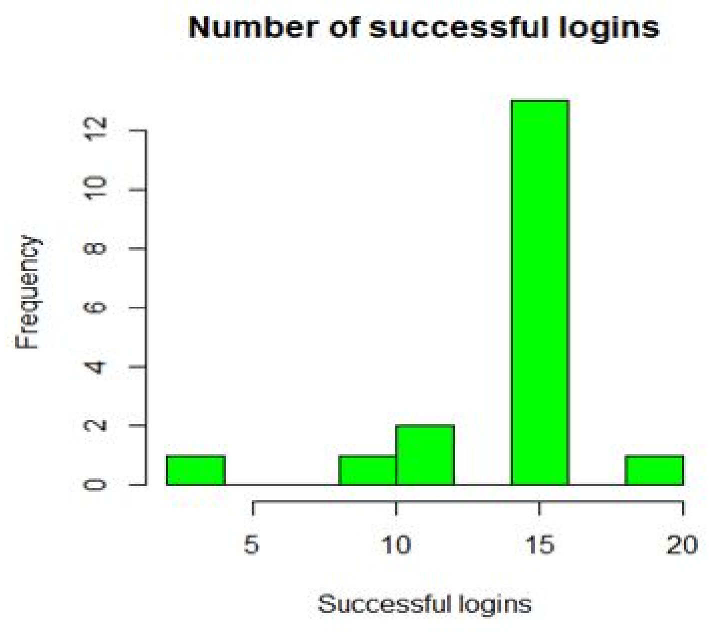


Figure 6 - Histogram for number of successful logins (Text21 scheme)

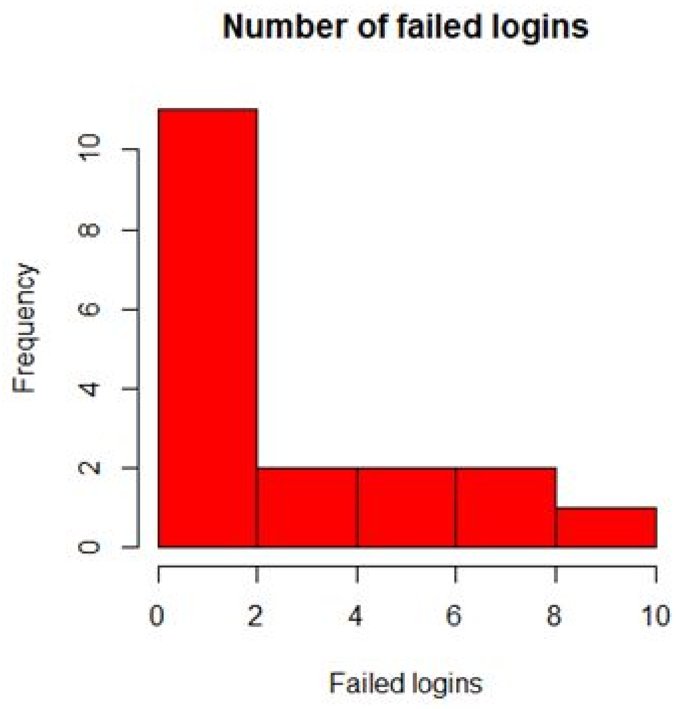


Figure 7 - Histogram for number of failed logins (Text21 scheme)

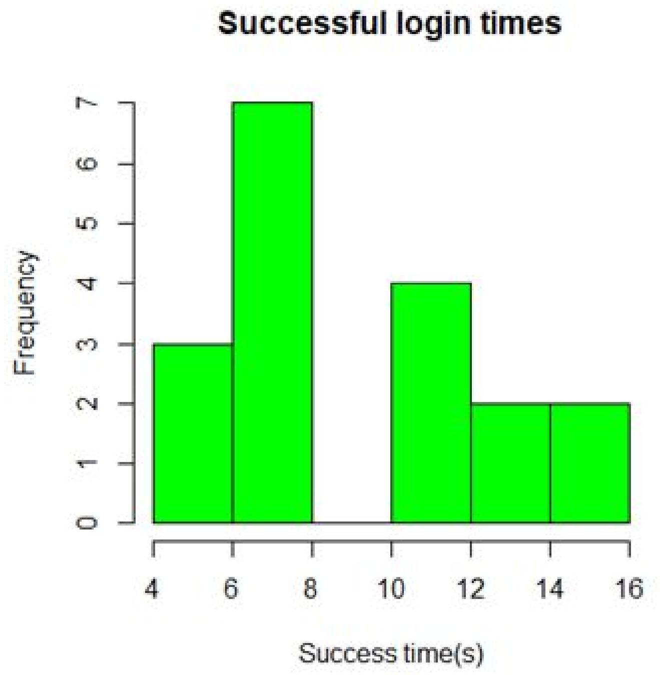


Figure 8 - Histogram for successful login times (Text21 scheme)

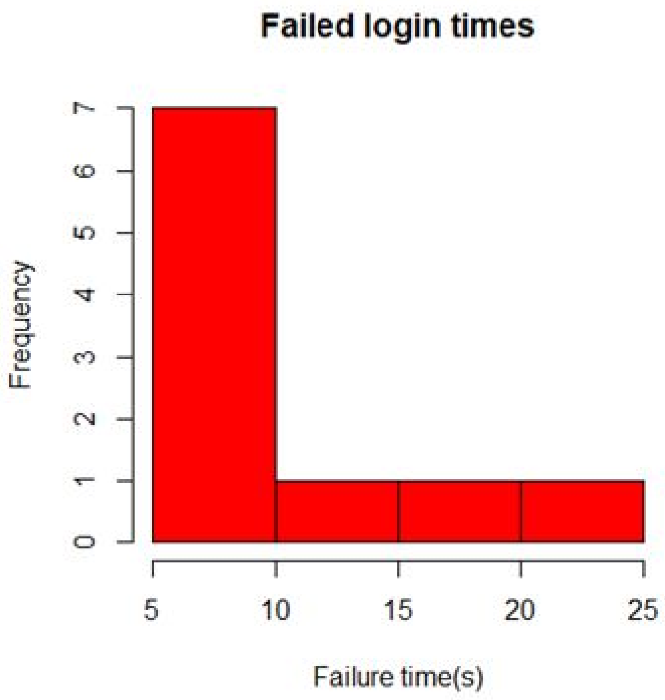


Figure 9 - Histogram for failed login times (Text21 scheme)

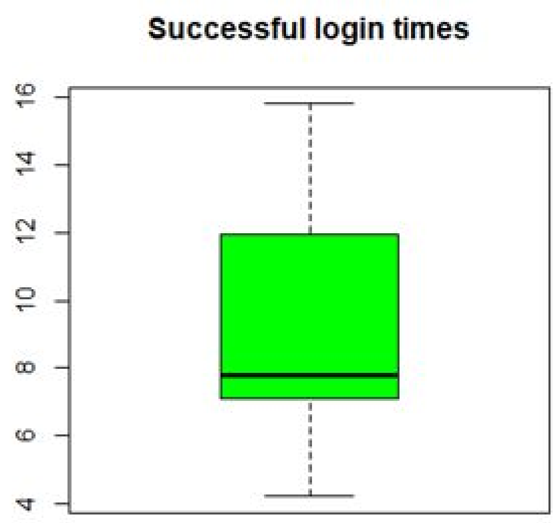


Figure 10 - Boxplot for successful login times (Text21 scheme)

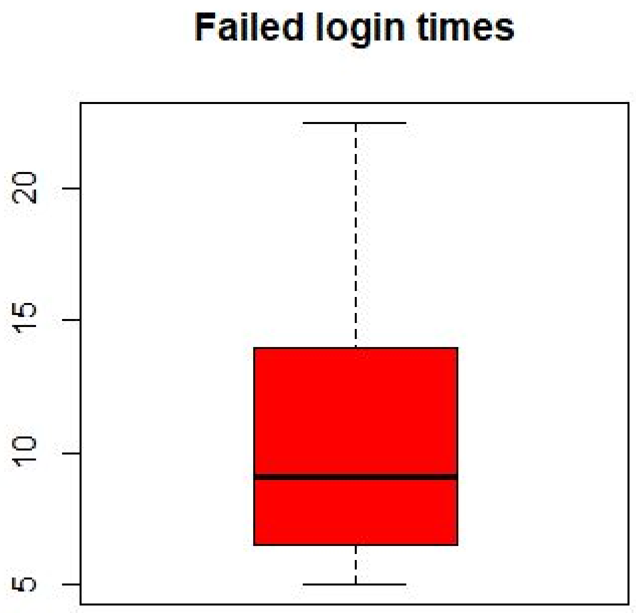


Figure 11 - Boxplot for failed login times (Text21 scheme)

Interpretation of the Text21 scheme statistics and graphs

In the above table (Table 1- Number of logins per user) we can conclude the following:

- There were 18 users in total.
- 8 users got 100% successful login rate.

The graphs that represent this table (Table 1) are figure 5, figure 6, figure 7. From these graphs we can conclude the following:

- Most users logged in a total of 10-15 times.
- Most users successfully logged in 14-15 times.
- Most users failed the login 0-2 times.

In the above table (Table 2) Mean, median and standard deviation for logins, we can conclude the following:

- The mean total login attempts is 16.5.
- The standard deviation for successful logins is 3.45.
- The standard deviation for failed logins is 3.32 meaning the resulting number of failed logins are not as consistent between failed logins.
- The standard deviation for total logins is 4.87 meaning some users logged in more or less than the mean total login amount.
- The average user succeeded logins 84.54% of the time.

In the above table (Table 3) Successful login times per user, we can conclude the following:

- Only 8 out of 18 users had a standard deviation below 3.
- The rest were between 3 and 14, meaning the success login times were not consistent.

In graphs that represent this tables are figure 8 and figure 10. From these graphs we can conclude that it takes the average user between 6-8 seconds to login successfully.

In the above table (Table 4) Failed login times per user we can conclude the following:

- There’s 1 user who has a standard deviation below 3
- The rest of the users have standard deviation between 3-21, which is not consistent at all.
- There are 3 users who have no standard deviation, because their mean and median are the same.

The graphs that represent this table are figure 9 and figure 11. From these graphs we can conclude the following:

- It takes the average user between 5-14 seconds for failed logins.

The times should be similar to the successful login times. This seems like failed login attempts were because the user did not know or even tried to guess and tried to login without typing much of a password.

1.3.2 Imagept21 Data

User	Successful Logins	Failed Logins	Total Logins	Success %
ipt101	16	16	32	50.00
ipt104	12	0	12	100.00
ipt105	15	7	22	68.18
ipt106	14	6	20	70.00
ipt109	15	0	15	100.00
ipt110	15	4	19	78.95
ipt113	17	12	29	58.62
ipt119	16	2	18	88.89
ipt131	12	3	15	80.00
ipt133	14	1	15	93.33
ipt134	15	4	19	78.95
ipt136	15	3	18	83.33
ipt137	16	4	20	80.00
ipt143	14	3	17	82.35
ipt145	14	1	15	93.33

Table 5 - Number of logins per user

Logins	**Mean**	**Median**	**Standard Deviation**
Success	14.66667	15	1.397276
Failure	4.4	3	4.436859
Total	19.06667	18	5.338093

Table 6 - Mean, median and standard deviation for logins

Mean Success% = 76.92

User	**Mean**	**Median**	**Standard Deviation**
ipt101	11.375	11	2.825479
ipt104	21.91667	21.5	6.625822
ipt105	9.133333	7	6.801961
ipt106	12.07143	11.5	4.730658
ipt109	23.13333	17	12.6314
ipt110	11.53333	11	2.899918
ipt113	10.52941	5	16.08617
ipt119	14.1875	11	13.2676
ipt131	37.41667	34	14.87193
ipt133	14.78571	15.5	3.826599
ipt134	21.26667	21	10.15218
ipt136	12.46667	11	3.62268
ipt137	19.5625	19	5.621017
ipt143	12.92857	9	10.60816
ipt145	25.14286	24	5.8554

Table 7 - Successful login times per user

User	**Mean**	**Median**	**Standard Deviation**
ipt101	28.125	20.5	20.90574
ipt105	14.71429	14	7.454625

ipt106	9.5	9.5	3.271085
ipt110	41.5	19.5	49.84309
ipt113	8.833333	10.5	3.713203
ipt119	8.5	8.5	0.707107
ipt131	45.66667	52	13.6504
ipt133	6	6	NA
ipt134	24.25	24	5.315073
ipt136	17.33333	16	3.21455
ipt137	22	16	12.67544
ipt143	15.66667	14	6.658328
ipt145	34	34	NA

Table 8 - Failure login times per user

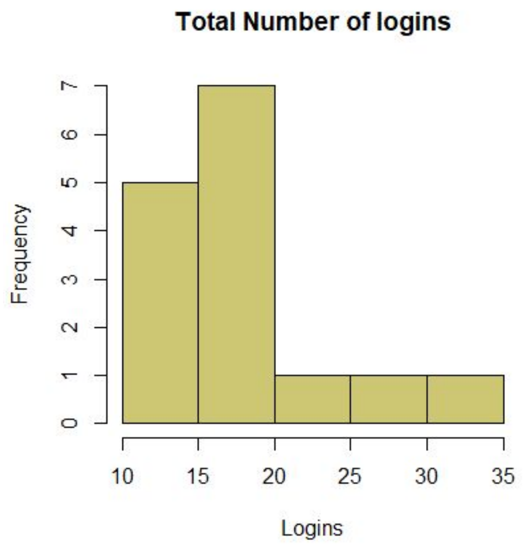


Figure 12- Histogram for total number of logins (Imagept21 scheme)

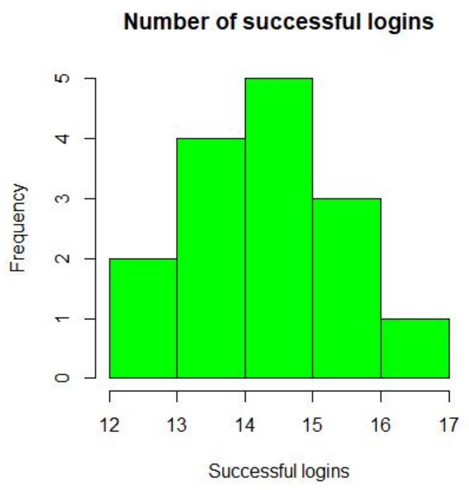


Figure 13- Histogram for number of successful logins (Imagept21 scheme)

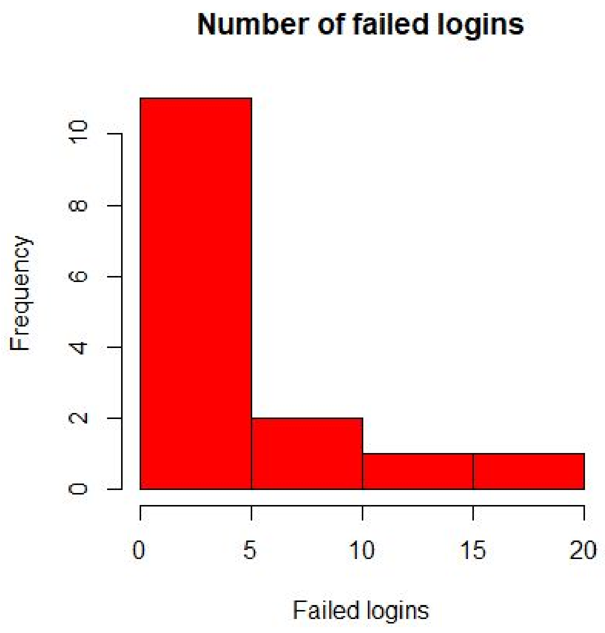


Figure 14 - Histogram for number of failed logins (Imagept21 scheme)

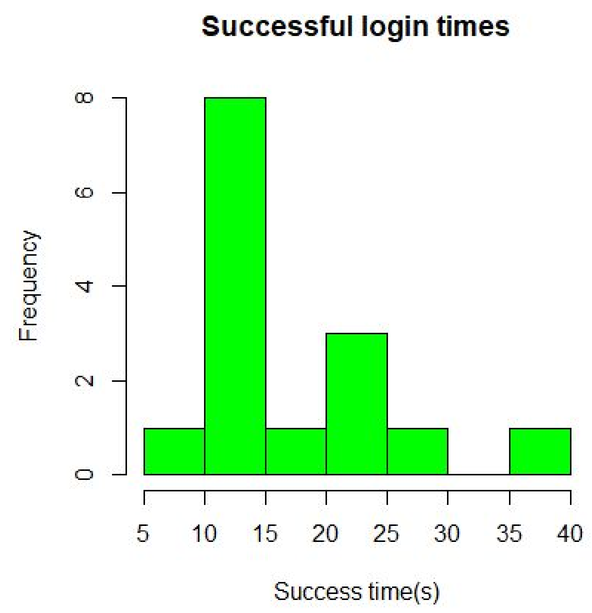


Figure 15- Histogram for successful login times (Imagept21 scheme)

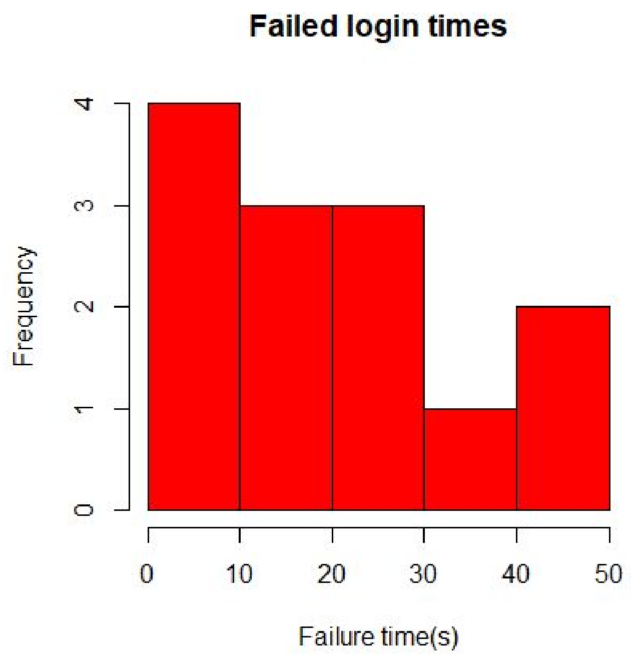


Figure 16 - Histogram for failed login times (Imagept21 scheme)

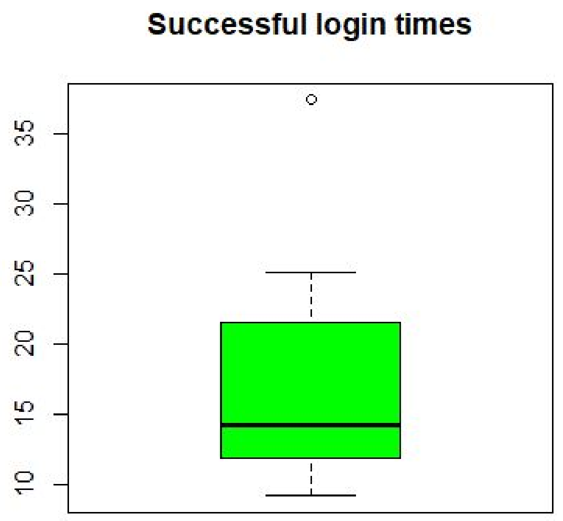


Figure 17 - Boxplot for successful login times (Imagept21 scheme)

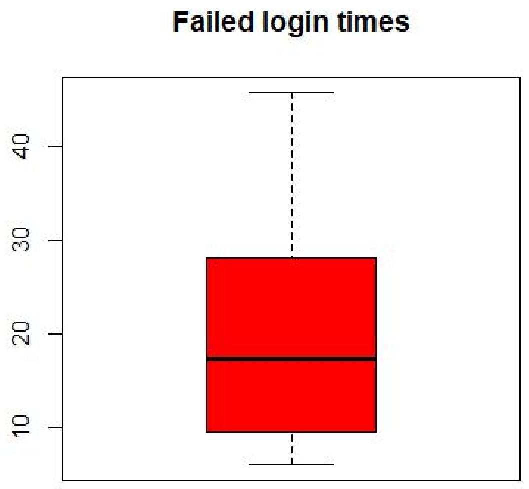


Figure 18 - Boxplot for failed login times (Imagept21 scheme)

Interpretation of the Imagept21 scheme statistics and graphs

In the above table Number of logins per user we can conclude the following:

- There were 15 users total.
- Only 2 users had a success rate of 100%.
- Only 4 user failed more than 5 times.

The graphs that represent this table are figure 12, figure 13, figure 14. From the graphs we can conclude the following:

- Most users logged in a total of 10-15 and 15-20 times.
- Most users successfully logged in 14-15 times.
- Most users failed the login 0-5 times.

In the above table Mean, median and standard deviation for logins we can conclude the following:

- The mean total login attempts is 19.06.
- The standard deviation for successful logins is 1.39 meaning the resulting number of successful logins are pretty consistent between the users.
- The standard deviation for failed logins is 4.43 meaning the resulting number of failed logins are not as consistent between the failed logins.
- The standard deviation for total logins is 5.33 meaning some users logged in more or less than the mean total login amount.
- The average user succeeded logins 76.92% of the time.

In the above table Successful login times per user we can conclude the following:

- Only 5 users had a standard deviation below 5.
- The rest had a standard deviation between 5 and 17 meaning the success login times were not that consistent.

In graphs that represent this table are figure 15 and figure 17. From these graphs we can conclude that it takes the average user between 10-25 seconds to login successfully.

In the above table Failed login times per user we can conclude the following:

- Only 2 out of the 15 total users did not failed an attempt.
- There were 4 users who had standard deviation below 5 and the highest standard deviation was 49.84 meaning all failed login times were very inconsistent.

The graphs that represent this table are figure 16 and figure 18. From these graphs we can conclude that it takes the average user between 10-30 seconds for failed logins. The times are a little longer than the successful login times. This makes sense because the user would take more time trying to remember instead of just inputting the password.

1.3.3 Conclusion

We believe that the success rate and the time taken to enter the password are the two properties that make a password more usable. So, to conclude we will be looking at both properties for each provided scheme to determine which scheme has the best usability.

For the success rate, the Text21 scheme had a successful login rate of 84.54%. However, there were a few outliers with failed logins, but those didn’t really affect the login success rate. If we were to remove these outliers the successful login rate would automatically increase. Most users successfully logged in 14-15 times and 2 users had a failed login attempt using Test21 password scheme. 8 users scored a 100% successful login rate. For the Imagept21 scheme, the successful login rate was 76.92%. There was a total of 15 users who attempted to login. And out of those 15 users, only 2 scored a 100% successful login rate. The standard deviation for failed and total login attempts was much higher than the Text21 schemes. So, we can conclude that Text21’s data is correct with little error.

As mentioned earlier, the successful login rates were 76.92% (Imagept21) and 84.54% (Text21), which are very close. However, since there are some obvious outliers in Text21, the percentage would increase if removed. So, looking at the above stats, we concluded that the Text21 has a better successful login rate. This may be because Text21 is a text-based password scheme and we all know that a text-based scheme is a very common password scheme. The other image-based password scheme could have its success rates increased if users were more familiar with that scheme.

For the time taken to enter the password scheme, the text-based Text21 scheme took an average of 6-8 seconds to login successfully and 5-16 seconds for failed login attempts. With this information we concluded that the failed login times could not have been real login attempts as the times should have been longer because the user would have to think about the password then type the password. The Imagept21 scheme took an average of 10-25 seconds to login successfully and 10-30 seconds for failed login attempts. With this information we conclude that the failed login times for Text21 are useless as they are scattered and too low for a real failed login attempt. The only failed login attempt time that makes sense is for the Imagept21 scheme. However, the fastest time taken to enter the password scheme is with Text21. It is almost 3 times faster than the other scheme. Again, this is probably due to the fact that users are more familiar with this password scheme and typing on a keyboard is much faster than clicking the squared images.

We conclude that Text21 scheme has the highest success rate and the fastest login times. Hence, it has the better usability among the password schemes that we tested. This is probably because all users are familiar with this password scheme. If in the future Imagept21 scheme is used more, users may be able to remember their passwords better and enter them faster but currently Text28 is still the best usable password scheme.

2. Design, Implementation, Statistical Inference

2.1 The Scheme

Nowadays, we can see that when a software generates a password, it usually has no meaning, too random and too hard to memorize. So, our team came up with a solution that can basically encrypt a meaningful word into a password and allows a user to memorize it rather easily.

We have a list of meaningful words in our database that have the length from 5 to 8 characters, 1800 words to be exact. We created a code to randomly select a word from the list, then convert the appropriate characters of the word to a special character by using our following method:

- ‘a’ will be converted to --> @
- ‘b’ --> 8
- ‘i’ --> !
- ‘o’ --> 0 (number 0)
- ‘s’ --> $
- ‘t’ --> +

To strengthen the converted password, we add 2-digit number of the current day. For example, if the word “tower” is selected as a password for the current user, it will be converted to “+0wer03” with 03 being the date this scheme was created, that is April the 3rd.

We believe that this scheme has a good usability and the password the will generated through this scheme will be easy for user to recall. Our password uses 20 normal characters in alphabet (omit the 6 characters that convert to the special characters), with 10 digits number, and 4 special characters, the total of characters is 34. We have at least 7 characters in the password (5 characters from the word lists plus 2 numbers), so the password space would roughly equal to:

$$\boxed{\text{34 characters, length 7} \Rightarrow \text{Log}_2 34^7 = 36 \text{ bits}}$$

Since the word selected for the password would be meaningful, The password would be easy to memorize, recall and reuse. Plus, with the length and the combination of the characters, the password provides a good security compared to other normal passwords. Based on Kaspersky website, the password “+0wer03” takes 2 hours to brute force with an average home computer. The word, selected for the password, will be then removed from the word list and only the user who selected that password will know what the word is.

2.2 Implementation of a password scheme

Extract Release.zip file from the folder provided and run PasswordSystem.exe

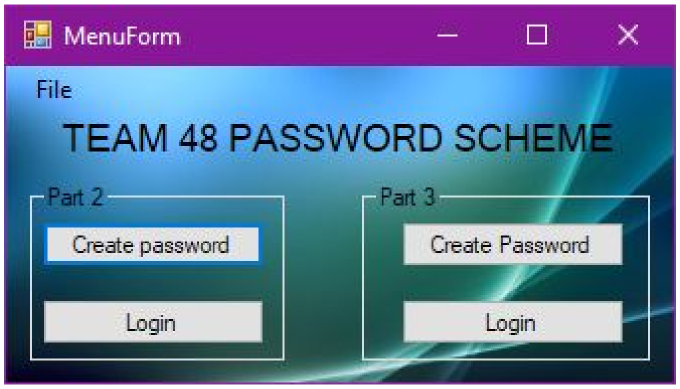
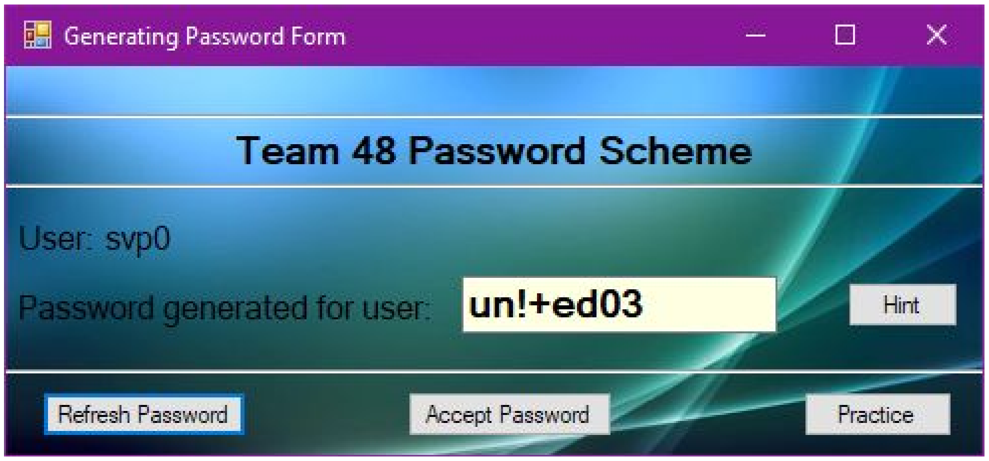


Figure 19: The first menu when running the software.

Figure 19 shows how the scheme has 2 parts that we called part 2 and part 3 for now. Part 2 is used for creating and login one password, and part 3 is used for creating and logging 3 passwords for emails, banking and shopping. The File menu is used to display Log file and account files (this is for the programmer's only).

When a user clicks "Create Password" button in Part 2, the following form below will appear:



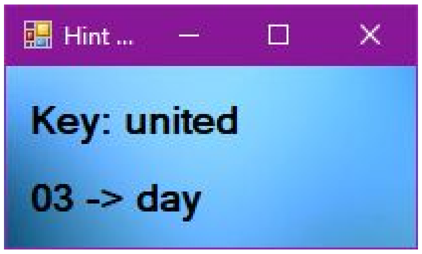
User: svp0

Password generated for user: un!+ed03

Figure 20: Generating Password Form.

The above screenshot demonstrates how this scheme will automatically assign the new username for user, in this case "svp0". We have 4 buttons and 1 password textbox. The "Refresh Password" button is used to generate new password; the user can press this button multiple times until they find the favorite password. The password will be displayed in the textbox and user can select and copy it but cannot modify it. The “Hint” button will open Hint form, “Practice” button will open a practice form, and finally, “Accept Password” button will save this password for this user to the database and then the form will close.

If user clicks ‘Hint” button, it will open the form below:



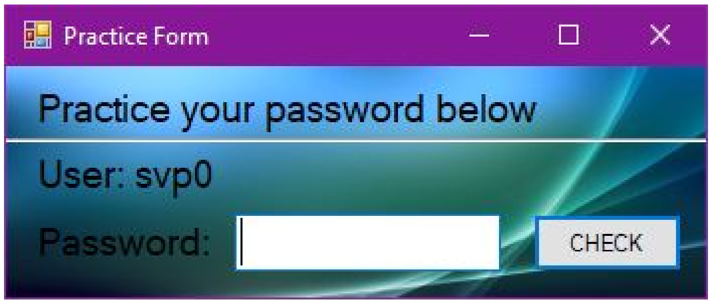
Key: united

03 -> day

Figure 21: The hint form appears.

The key is basically the word that we used to convert to the password. It helps user to memorize the password easier. The second line display the number of the day.

If user clicks “Practice” button in Generating Password form, it will open the form below:



Practice your password below

User: svp0

Password:

Figure 22: The practice form

In the above Picture, we simply showed 3 things here. The username, the textbox for user to input their password for practice, and the “Check” button to check the correctness. It can display the message if user input right or wrong password.

Finally, click “Accept Password” button in Generating Password form to save this and close the form. User should now click “Login” button in Part 2 in Menu Form to login. The new login form will appear below:

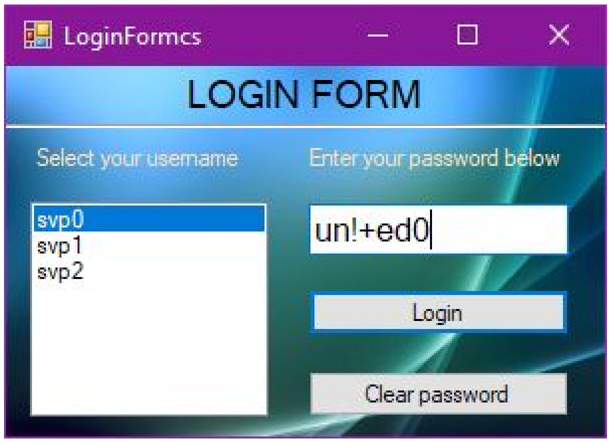


Figure 23: Login Form.

To provide the simplicity and convenience for us, we created the List Box to the left and add all the available usernames from our database to the box. User simply choose their username, then enter the password to the textbox on the right, then click “Login” button to attempt a login. “Clear password” will clear the text in the password box.

2.3 Version 2 of the new password scheme

Extract Release.zip file and run PasswordSystem.exe

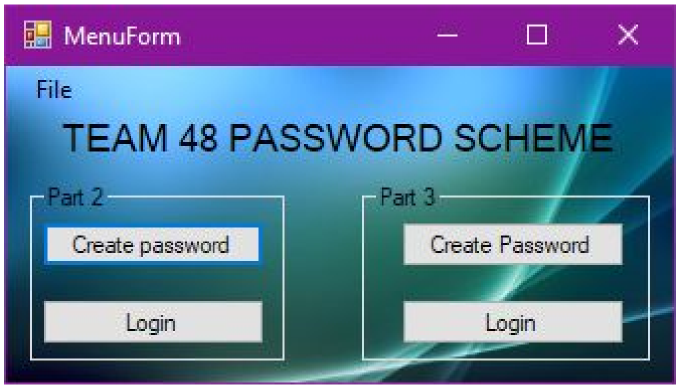
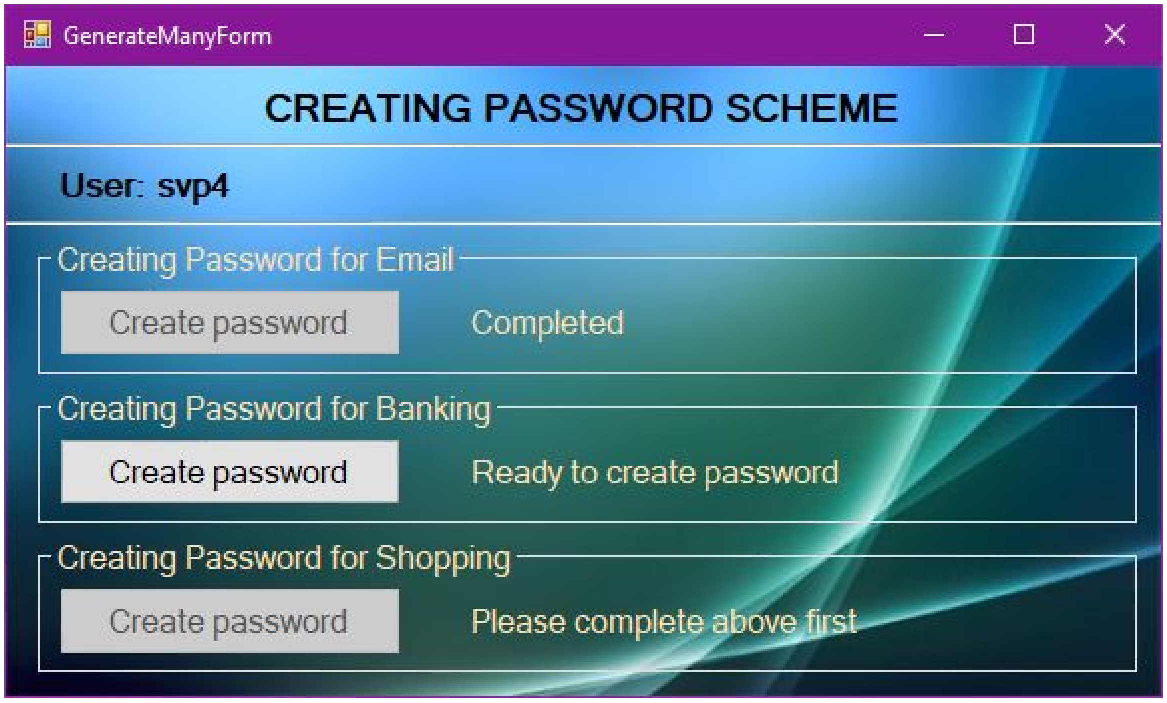


Figure 24: The first menu when running the software.

Our scheme has 2 parts that we called part 2 and part 3 for now. Part 2 is used for creating and login one password, and part 3 is used for creating and logging in 3 passwords for email, banking and shopping. The File menu is used to display Log file and account files (This is for programmer's only)

For this part, user should click "Create Password" in part 3:


GenerateManyForm
CREATING PASSWORD SCHEME
Creating Password for Email
Create password
Creating Password for Banking
Create password
Creating Password for Shopping
Create password

User: svp4

Completed

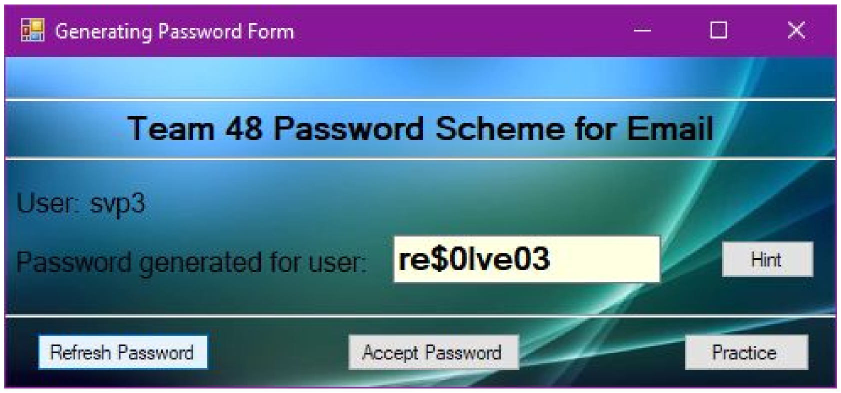
Ready to create password

Please complete above first

Figure 25: GenerateMany Form

Figure 25 shows how the scheme will automatically assign new username for the user. There are 3 parts: Email, Banking and Shopping. Each part has 1 button and 1 status label on the right. Initially, 2 other “Create Password” buttons will be disabled and would only work if user completes the current(running) process.

If user clicks “Create Password” for the part Email, the Generating Password Form will appear.



Team 48 Password Scheme for Email

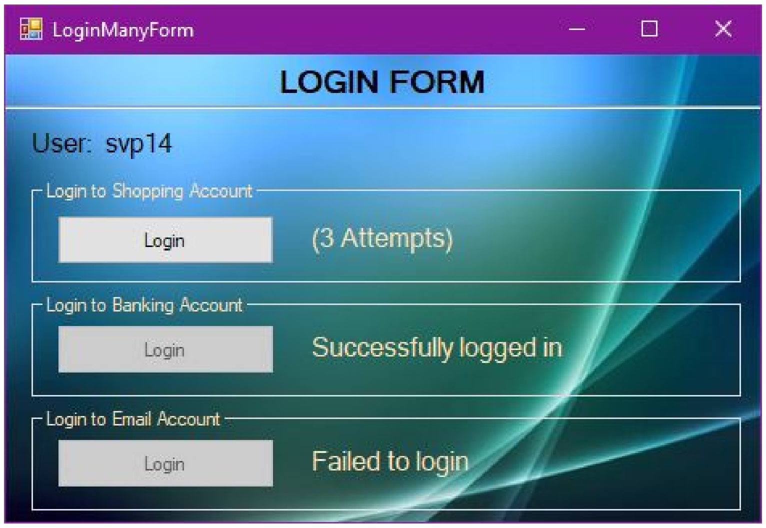
User: svp3

Password generated for user: re$0lve03

Figure 26: Generating Password form for Email.

This form is exactly the “Generating Password form” as shown in part 2, we simply changed the title and added the password and account to different database file. All the buttons are the same as we reuse the form.

The “Banking” and “Shopping” password forms are exactly like the “Email form, we only changed the title and added the saving password to the different database file. Once the user completes all the password, the form will close automatically. Now, user should click “Login” button in Part 3 in Menu Form to display this form below:



LOGIN FORM

User: svp14

Login to Shopping Account — (3 Attempts)

Login to Banking Account — Successfully logged in

Login to Email Account — Failed to login

Figure 27 - Email, Shopping and Banking login page

Figure 27: The 3 parts “Email”, “Banking” and “Shopping” will be shuffled each time user opens it. Similar to Creating Many form, each part has a button and a status. Each status will display the number of attempts user has; initially, it displays 3 attempts and will reduce whenever user provides a wrong password. If it reaches zero, it will display “Failed to login”, and if user manages to login successfully in 3 attempts, it will change to “Successfully logged in”.

If user clicks “Login” button in Shopping part, the Login Form will appear:

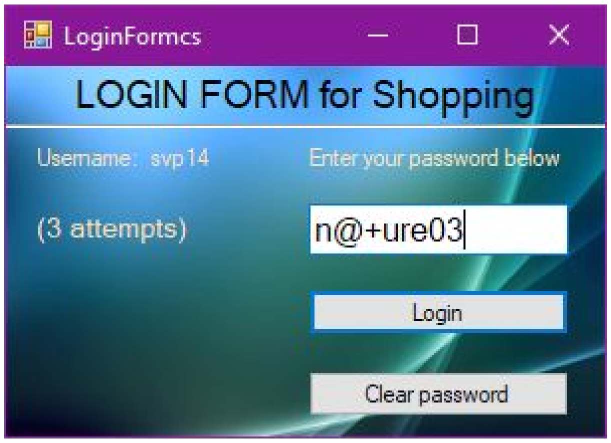


Figure 28 - Entering the password

Figure 28: For this part, we removed the list box and passed the username in LoginMany form to this form with the status “(3 attempts)”. The “Login” will get the password and check in our database; if the password is correct, it will close the form and change the status in LoginMany form, if it is wrong, the status will be changed to “(2 attempts)” and allow user to re-enter the password. When there is no more attempt, the form will be closed and disabled the “Login” button in LoginMany form. “Clear password” button will clear the text in the box.

The source code of the above Password scheme is accessible through the link below:

- https://github.com/ThuongMai3004/PasswordSystem.git

2.4 Questionnaire

Please find the Team48-survey.pdf file in the compressed(.zip) folder. Link to the actual survey is provided below.
https://hotsoft.carleton.ca/comp3008limesurvey/index.php/579233?lang=en

We have also provided a PDF of the questionnaire in our Submission folder.

2.5 Testing

Testing was successfully completed, please refer to next section (2.6) for results and comparison. Raw data is also available in .zip folder if it is required, please refer to README.txt file for all files and folder and its contents.

2.6 Comparison

2.6.1 Team 48 password scheme data

The data tabulated below was collected from the users testing our password scheme, and was processed using R using the same code as provided above in section 1.2.

user	success	failure	total	success %
svp01	5	2	7	71.43
svp02	8	0	8	100
svp03	5	2	7	71.43
svp04	10	0	10	100
svp05	6	2	8	75
svp06	1	5	6	16.67
svp07	6	3	9	66.67
svp08	4	3	7	57.14
svp09	5	2	7	71.43
svp10	6	1	7	85.71

Table 9 - Users' Data (Team 48 Password Scheme)

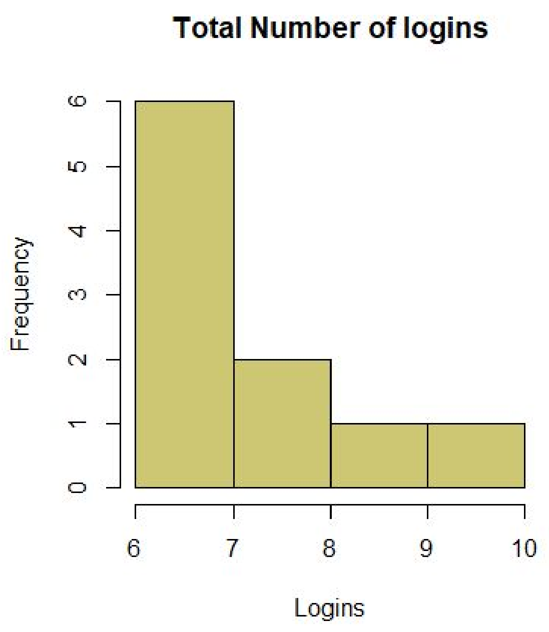


Figure 29 - Number of Logins (Team 48 password scheme)

Total attempts	Successful attempts	Failed attempts	Percent success	Percent fails
76	56	20	73.68421053	26.31578947

Table 10 - Simplified User Data (Team 48 Password Scheme)

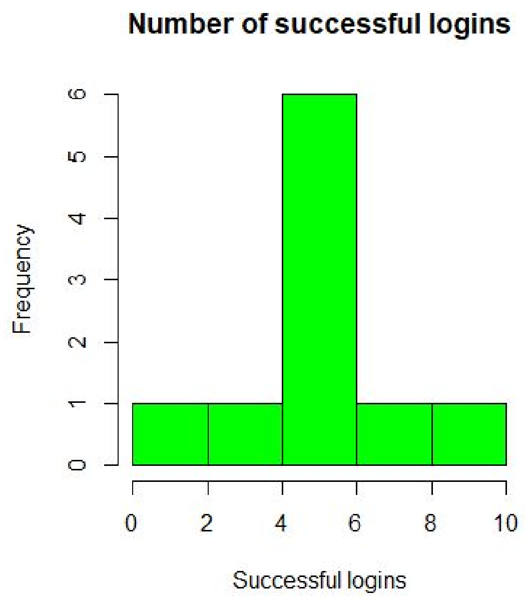


Figure 30- Number of successful logins (Team 48 Password Scheme)

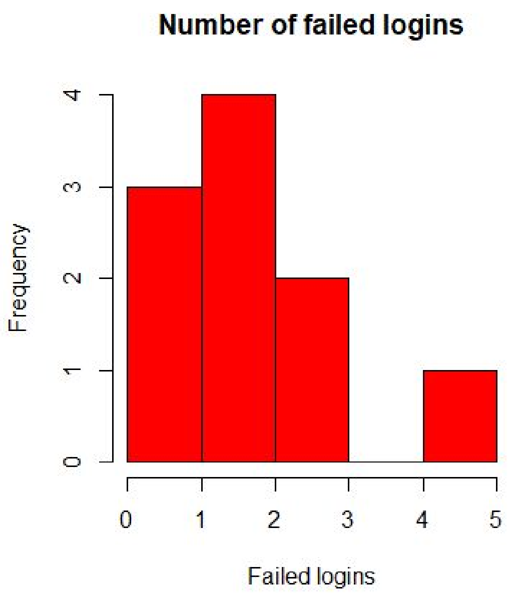


Figure 31- Number of failed logins (Team 48 Password Scheme)

Referring to logins and simplified logins table provided above, there were a total of 10 participants that made a total of 76 login attempts. 56 out of 76 were logged as successful attempts and 20 as failed attempts, which gives us a success and failure rate of 73.68% and 26.31%, respectively. Now comparing it to Text21's login data, which has successful and failed login rate of 84.54% and 15.46%, respectively. Therefore, we can conclude that Text21 has a better rate of successful logins compared to Team 48's password scheme.

logins	mean	median	sd
success	5.6	5.5	2.366432
fail	2	2	1.490712
total	7.6	7	1.173788

Table 11 - Mode-Median-SD (Team 48 Password Scheme)

The mean, median, and standard deviation for success is 5.6, 5.5, and 2.36, respectively. Similarly, the mean, median, and standard deviation for failure is 2, 2, and 1.49, respectively. Therefore, by combining both success and failure mean, median, and standard deviation, we get a total of 7.6, 7, and 1.17, respectively.

user	mean	median	sd
svp01	13.9588	12.015	10.23438
svp02	11.727625	10.1695	7.734513
svp03	8.4788	7.498	4.052038
svp04	6.2385	5.61	4.209026
svp05	5.03283333	5.3345	2.766821
svp06	9.055	9.055	NA
svp07	12.2253333	6.07	18.56744
svp08	8.5345	8.4555	2.385138
svp09	12.5362	14.831	4.260217
svp10	13.765667	14.503	9.870088

Table 12 - Success Login Times (Team 48 Password Scheme)

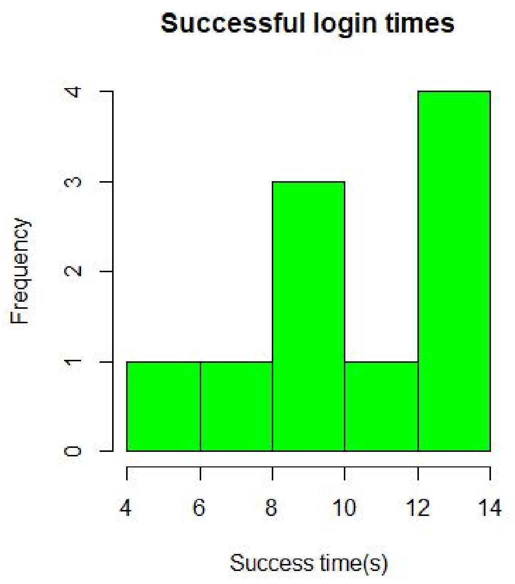


Figure 32 - Successful login times bar graph (Team 48 Password Scheme)

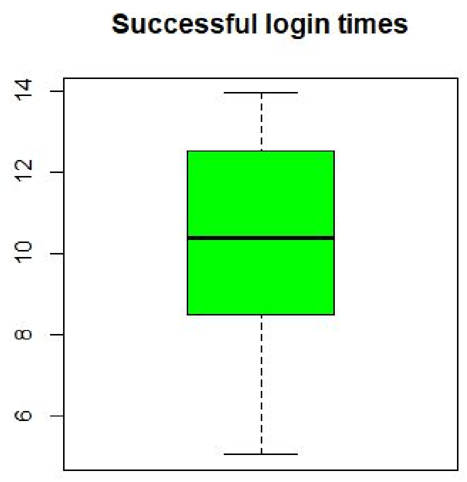


Figure 33- Successful logins times Boxplot (Team 48 Password Scheme)

Average mean	Average median	Average sd
10.1553258	9.35415	7.119961

Table 13 - Average successful login times (Team 48 Password Scheme)

The average mean and median time for successful login times are 10.15s and 9.35s, respectively. The average standard deviation is 7.11, comparing it to what we have per user, only one user is in one standard deviation, rest are in 2 or 3 standard deviation. Now comparing this data with Text21's table of success times, the average mean and median time it takes to login successfully is about 9s and 7s, respectively. Therefore, we can conclude that Text21 password scheme requires shorter time to successfully login compared to Team 48's password scheme.

user	mean	median	sd
svp01	4.1685	4.1685	4.652056
svp02	NA	NA	NA
svp03	11.724	11.724	11.7408
svp04	NA	NA	NA
svp05	3.2485	3.2485	2.020204
svp06	5.7184	2.768	6.263056

svp07	10.821667	2.048	16.38746
svp08	3.003	3.303	1.346306
svp09	8.3035	8.3035	10.96228
svp10	9.104	9.104	NA

Table 14 - Failed Login Times (Team 48 Password Scheme)

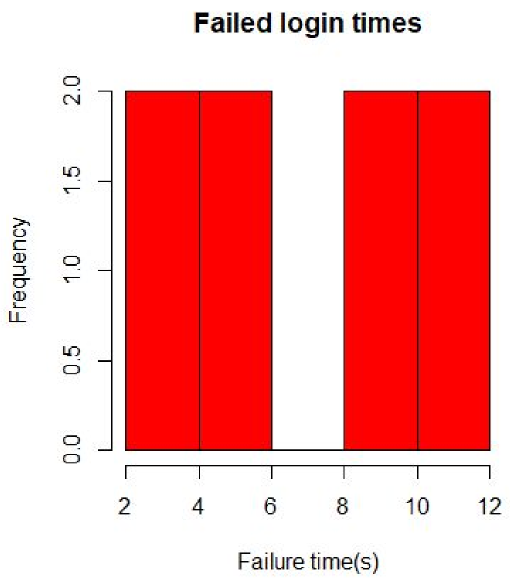


Figure 34- Failed login times bar graph (Team 48 Password Scheme)

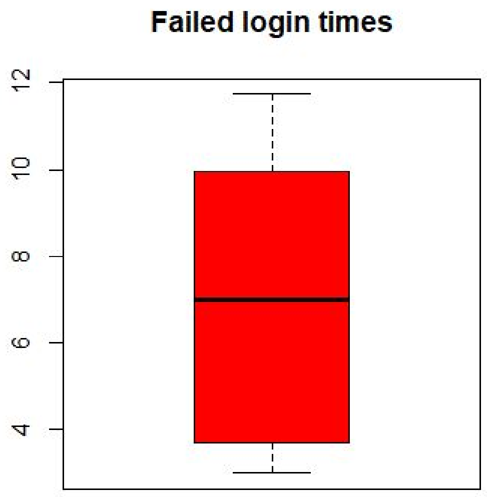


Figure 34 - Failed login times Boxplot (Team 48 Password Scheme)

Average mean	Average median	Average sd
7.0114458	5.5834375	7.624594

Table 15 - Average failed login times (Team 48 Password Scheme)

The average mean and median time for failed logins are 7.01s and 5.58s, respectively. The average standard deviation is 7.62, comparing it to what we have per user, only one user is in one standard deviation, rest are in 2 or 3 standard deviation. Now comparing this data with Text21's table of failure time, the average mean and median time it takes to unsuccessful logins are about 8s and 7.5s, respectively. Therefore, we can conclude that users are likely to fail login if they spend less time than average successful login time calculated above.

2.6.2 Discussion

Quantitatively, comparing the statistics of password schemes Text21 and Team48, results are insignificant. The successful login attempts using Team 48's password scheme have been reduced by ~11%, yielding ~11% increase in failed attempts as compared to Text21 password scheme. Similarly, the average amount of time it requires to login has also increased by ~2s using Team 48's password scheme. However, the results are logical and expected. Since Team 48's password scheme was designed to be more secure because our password scheme implements substitution model for some alphabets to be replaced with special characters, resulting in more combinations and has about 32-bits of password space. Our log files generated by the software indicated that users are likely to make fail attempts in the beginning due to improper use of special characters and gradually improve the successful login attempts. Therefore, it causes users to make slightly more failed attempts and takes slightly longer to login.

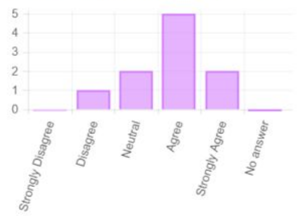
Qualitatively, users' perception about Team 48's password scheme was overall positive according to feedback received through surveys. To answer the statement if users would prefer Team 48's password scheme over user-chosen password scheme, 30% of the users stayed neutral, 40% agreed, and 30% strongly agreed. Similarly, to answer the statement if it was easy to use Team 48's password scheme, 20% of the users chose to stay neutral, 50% agreed, and 30% strongly agreed. Therefore, we can conclude that Team's password scheme is simple, effective, and easy to use. Please refer to appendices section for survey results and its bar graphs.

2.6.2 Conclusion

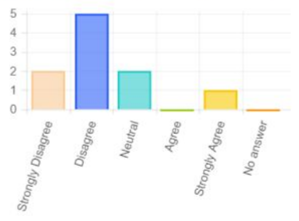
Finally, Team 48 successfully delivered the goal of providing a secure and easy to memorize (meaningful) password scheme. Even though our scheme makes use of dictionary words that are identified as old-fashioned user-chosen passwords that are vulnerable to attacks and are also easily crackable, usually under an hour. Our scheme re-enables the use of those dictionary words that are easy to recall but now has more strength and requires at least 4 times as much time compared to plain-text words to crack. For example, a password ‘volatile03’ is crackable in 3 hours, according to Kaspersky lab. However, using our password scheme, the same password represented as ‘v0l@+!le03’ and it now takes 2 days to crack, which shows significant increase in strength. Furthermore, this scheme will help users to generate unique passwords for each platform/website they use. According to our survey results, majority of the users agreed to have same passwords on different platforms, which means in the event of security being compromised on any of the websites, other websites used by the user will also likely be accessible to attacker using same credentials that can jeopardize their personal information available on these websites.

Appendices

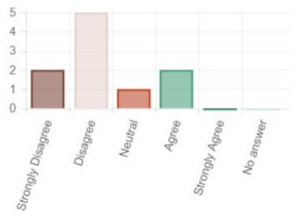
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Is it easy to recall your chosen passwords?]



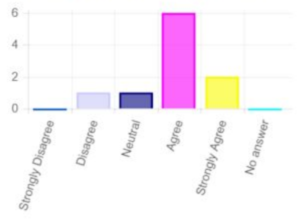
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Is it hard to recall your chosen passwords?]



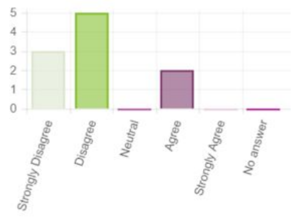
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Is it easy for you to think of a new password for any new platform you sign up for?]



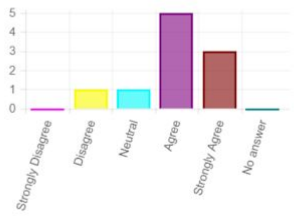
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Is it hard for you to think of a new password for any new platform you sign up for?]



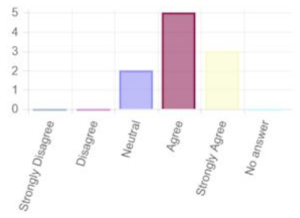
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Is it easy for you to choose a unique password for each new platform you sign up for?]



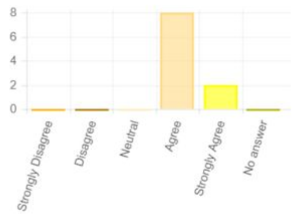
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Is it hard for you to choose a unique password for each new platform you sign up for?]



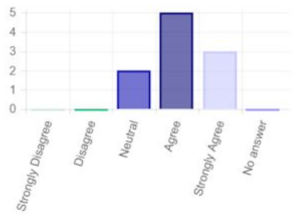
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Is it likely that you have a same password on multiple platforms?]



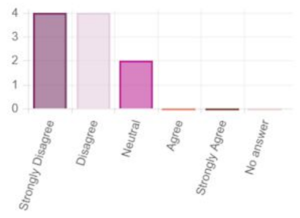
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Did team48's password scheme help you choose a unique password with one click?]



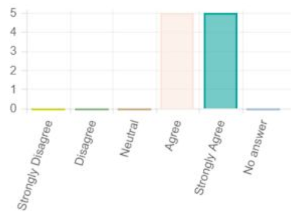
Please provide your responses for each statements given below about Team 48's Password Scheme.
[Did you find team 48's password scheme easy to use?]



Please provide your responses for each statements given below about Team 48's Password Scheme.
[Did you find team 48's password scheme hard to use?]



Please provide your responses for each statements given below about Team 48's Password Scheme.
[Did the hint feature help you memorize the password easily?]



Please provide your responses for each statements given below about Team 48's Password Scheme.
[Would you prefer Team48's password scheme over user-chosen text passwords?]

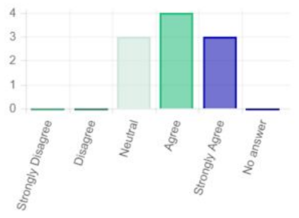


Figure 35 - Survey results with bar graphs (Team 48 Password Scheme)