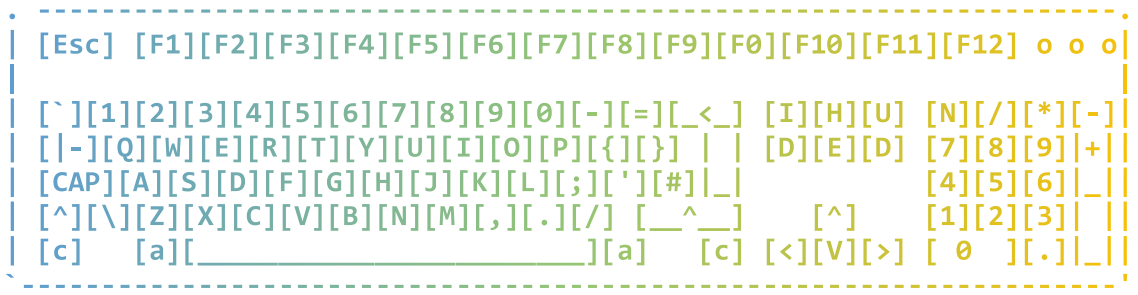
Rhythmic Keylogger for Authentication Report

CS4203: Assessment 2, 2021-22

Matriculation number: 190021081

Word count (not including figures, references and appendices): 3164



1. Introduction

In this report I first briefly introduce the concept of keystroke dynamics, before explaining two issues with their use for authentication (that I sought to investigate) in Section 2. I then provide further background into the field of study, highlighting two recent investigations into the practicality of their use in authentication in Section 3. In Section 4 I formalise my two hypotheses and explain how I investigated them, before analysing the results I obtained in Section 5. In Section 6 I briefly discuss possible avenues of further experimentation, and then conclude in Section 7.

Snippets of my code that perform keylogging are included in Appendix A. My full implementation is included in my .zip submission.

A sampling of my (processed) experimental data is included in Appendix B. My full dataset is included in my .zip submission.

2. Problem Statement

"Keystroke dynamics" refers to the profiling of how an individual types, computed with data from appropriate keylogging. They are a potential method for user authentication. However, I believed them to have some shortcomings for use in two-factor authentication in conjunction with a username/password pair, and designed experiments to test my theories.

My first intuition was that whilst using a chosen rhythm might make it easier to log in authentically, it would also draw attention to an onlooker that keystroke dynamics are part of the authentication, as well as making it easier for an adversary to mimic someone else's keystroke dynamics. My second intuition was that small factors such as minor injury or even a change in office furniture could sufficiently disrupt a user's typing to prevent them being able to authenticate.

3. Relevant Background

When a user attempts to authenticate, there are four possible outcomes:

- A True Positive (TP): the user is who they say they are, and authenticate successfully.
- A False Negative (FN): the user is who they say they are, but fail to authenticate.
- A False Positive (FP): the user is not who they say they are, but authenticate successfully.
- A True Negative (TN): the user is not who they say they are, and fail to authenticate.

3.1. Issues in the Field

The use of keystroke dynamics-based authentication and identification is not new, and arguably has its beginning in radiotelegraph communication; during World War II, British radio interceptors were able to identify German operators by their "fist, the personal style of tapping out a message" [1].

Modern keystroke dynamics analysts can profile users in a number of ways, including logging the time between keystrokes, identifying which keys a user prefers (left or right shift, use or not of the numpad, etc), and recording the frequency of errors (most simply characterised by use of the backspace and delete keys) [2]. Profiling can be done with statistical algorithms (e.g. Bayesian classifier [2]) or (more recently) machine-learning can be used to both build a profile and authenticate against it.

In comparison to many alternatives for two-factor authentication (e.g. fingerprinting and other hard biometrics), keystroke dynamic recognition does not require additional hardware. Additionally, it can be implemented such that a user does not need to actively engage with it (unlike, for example, receiving a one-time PIN on a secondary device). Both factors can make it a desirable choice for two-factor authentication, provided that it is found to be effective (minimising FNs and FPs).

3.2. Recent Work

Adesina and Oyebola tested the impact on functionality of user age group and gender. They found that the demographic of a user did not have significant impact on the utility of keystroke dynamics-based authentication in their experiment. Unlike previous work their experiments were on touch-screen mobile devices rather than physical keyboard input on desktop computers. Their findings were therefore supportive of keystroke dynamics in two-factor authentication for mobile devices, addressing concerns that its efficacy would be impacted by user demographic [3].

Çevik, Akleyek and Koç investigated the impact of implementation details for a keystroke dynamics profiler. They found that machine learning was more reliable than classical statistical tests, and that tree-based machine learning algorithms performed best, with an average accuracy score of 0.94. They conclude that that keystroke dynamics are "a viable alternative for two-factor authentication".

4. Methodology

I first formalised my two intuitions (discussed in Section 2) into two hypotheses, to be tested in corresponding experiments:

> Hypothesis 1. Both TPs and FPs will be more common when users (attempt to) conform to a specific rhythm.
>
> Hypothesis 2. The rate of FNs will increase (and TPs correspondingly decrease) when users are subject to minor impediments.

For the sake of my experiments I will only be considering the timing data for key presses. There are four ways to characterise this: the time between each key being depressed and released; the time between the release of one key and depressing the next; the time between the depression of one key and the next; and the time between releasing one key and the next [3].

In order to test my hypotheses, I coded a keystroke logger that records the time between character inputs to the command line: this is subtly different from true key press data as keys such as Shift are not logged, and it only records when character keys are released, not when they are depressed. In order to access the OS-level event listener to record all key presses and all stages of each key press in Java it appeared to be necessary to either implement a GUI (to which listeners can be attached), or use a platform-specific third-party library that can make calls to the OS. For the sake of my experiment I thought that character inputs would suffice, but due to Java's buffered input readers my first implementation attempt would record all keys as being pressed simultaneously. I did create an InputStreamReader subclass with a buffer size of one character, but the problem persisted, so rather than expend more time trying to work around Java's buffering I implemented my experimental data-gathering as Bash scripts.

The scripts output data to CSV files, which I then analysed with Microsoft Excel. Timing data is recorded for both the username and password, and characterised as the time in milliseconds between each character being input (corresponding to the key being released). Four volunteers took part, and all experiments were done on my Windows laptop via the Ubuntu Windows Subsystem for Linux (the scripts should run on lab machines, although the spreadsheet may not format correctly on LibreOffice Calc). I have a wired keyboard, which some volunteers noted has bulkier keys than they are used to – this might have impacted their keystroke dynamics but hopefully this is mitigated by the fact that all volunteers used it. If a volunteer misspelled an ID or password then it did not count towards their number of attempts or baseline inputs, and the mistake is omitted from my filtered experimental data.

Baselines were calculated as the median value (for each character) of 5 entries of an <ID, PW> pair by a volunteer. I used medians rather than means on the assumption that (at least) the first time a volunteer typed a new ID or password it would be anonymously slow, which would impact a mean more than a median. A successful authentication required 70% of timings in both the username and password being within a 50% margin of error of the baseline. I considered investigation of good tolerances to be an experiment in and of itself and as such simply chose values that obtained a workable number of TPs and TNs. I found that lower margins of error (e.g. 20%)

resulted in hardly any TPs, which would have made it very difficult to evaluate my results in regard to my hypotheses, which assume that an honest user without impediment can usually authenticate successfully, and that margins of error exceeding 50% created too many FPs.

4.1. Experiment 1

I defined 4 test cases (i.e. <ID, PW> pairs). Each test case was assigned to a single volunteer, from whom a baseline was determined for that <ID, PW> pair with their natural typing rhythm. A baseline for rhythmic input with a rhythm of their choice was calculated in the same way. To mimic the scenario I imagined that deliberate use of a rhythm could be noticed by a "shoulder-surfing" adversary, the volunteers watched one another enter the rhythmic baselines, allowing them knowledge of tempo as well as rhythm. For the natural rhythm they did not observe the creation of baselines, to mimic an adversary unaware of the use of keystroke dynamic-based authentication.

The assignment of volunteers to tests cases and their chosen rhythms was as follows:

Volunteer	Test Case	Username	Password	Chosen Rhythm
0	0	aaaaa	123456789	*ta-tum-ta-tum*
1	1	edward	wedfvbghyu	*Twinkle Twinkle Little Star*[1]
2	2	jc311	Password1	*The Can Can*[2]
3	3	MrBig	s3cr3t	*The Mii Channel Theme* from the Nintendo Wii[3]

I intended for all volunteers to then enter each <ID, PW> pair 3 times with their natural rhythm, and 3 times with the rhythm decided by the volunteer assigned to each pair. Due to a coding error they had 5 attempts to mimic the rhythmic baselines, but I will use proportional accuracy rates when analysing my results so this should not be a significant issue.

4.2. Experiment 2

I used the same 4 <ID, PW> pairs, but created a baseline for each pair for each user (i.e. 16 baselines) with natural rhythm and no impediment. The natural rhythm baselines for Experiment 1 are a subset of these baselines.

I then had volunteers attempt 3 times to authenticate against their own baselines for all <ID, PW> pairs with the following impediments:

- None (control test).
- First and second finger on each hand taped together – they could still use their fingers to type, but could not splay their fingers as one normally might whilst typing (to mimic a sprain or other minor injury).
- Typing using only their non-dominant hand (to mimic a more severe injury).
- Typing whilst standing up (to mimic use of a standing desks on some days).
- Typing with cold hands (volunteers held a bag of frozen peas for 1 minute prior to their set of attempts, to simulate attempting to log in after arriving at a building during cold weather).

I tried to ensure that the test cases utilised varying space on the keyboard: "edward" can easily be typed with one hand, and due to the path along the keyboard in typing "wedfvbghyu" it is potentially easier with one hand than with two, but "jc311" and "Password1" use keys on both sides of the keyboard, which I thought would increase the impact of reduced mobility. Further investigation of the effect between required range of motion and the impact of each impediment was unfortunately not within the scope of my experiments.

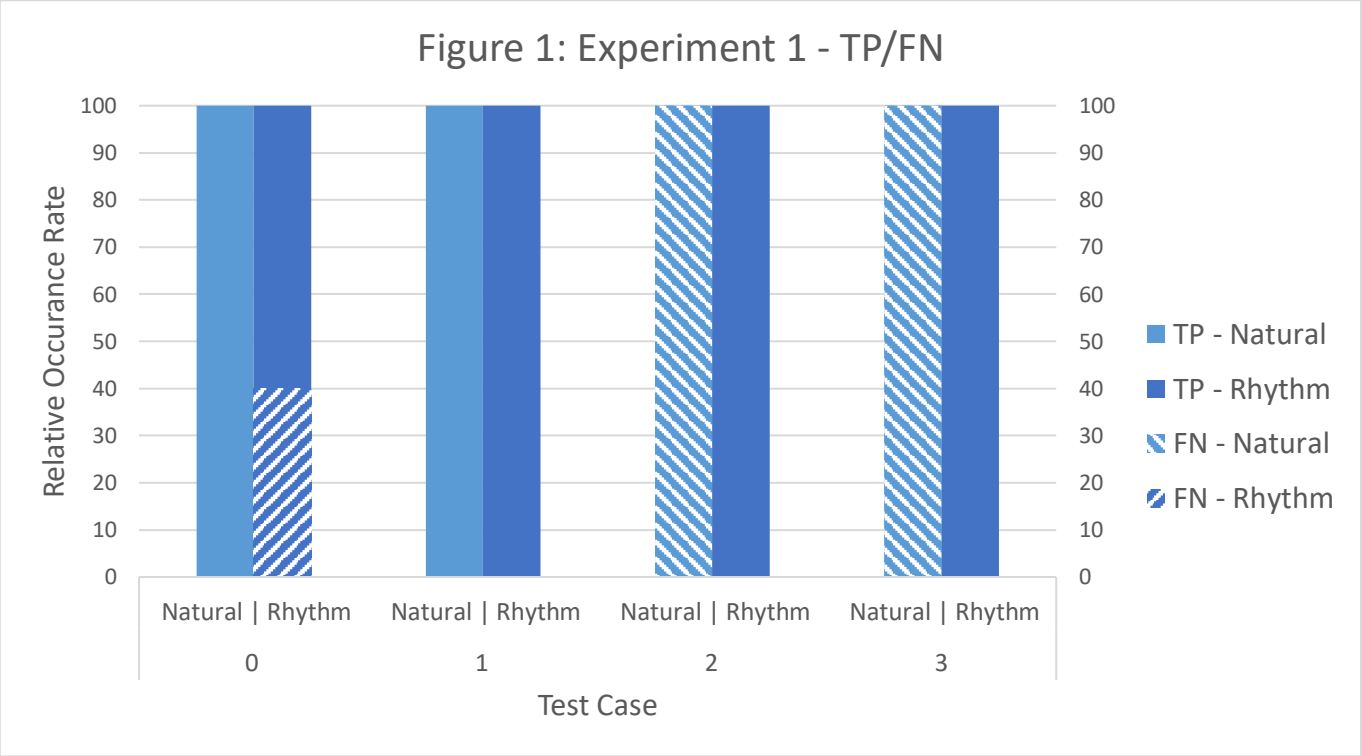
[1] https://www.youtube.com/watch?v=lgeRGN5qBj8&t=8s
[2] https://www.youtube.com/watch?v=T59EDTqqW0A&t=22s
[3] https://www.youtube.com/watch?v=bjDA-Jns51Y

5. Results

The striped portion of columns are essentially when the authentication fails: either a false negative or a false positive. So, where a bar is mostly solid, keystroke dynamics are behaving well as part of two-factor authentication. Whether striped denote FPs or FNs depends on the graph and is noted in the corresponding legend. Neighbouring stripes are at mirrored angles to aid visibility.

5.1. Experiment 1

To determine if using a chosen rhythm makes it easier for an honest user to successfully authenticate, I compared the proportion of attempts that resulted TPs against the proportion of FNs for each test case. For this set of results from Experiment 1, each user was assigned to a single test case and had 3 attempts to match the natural baseline and five to match their chosen rhythm, as mentioned. The results are presented in Figure 1, with TP:FN represented as stacked bar charts, with bars grouped by test case (and thus by volunteer).



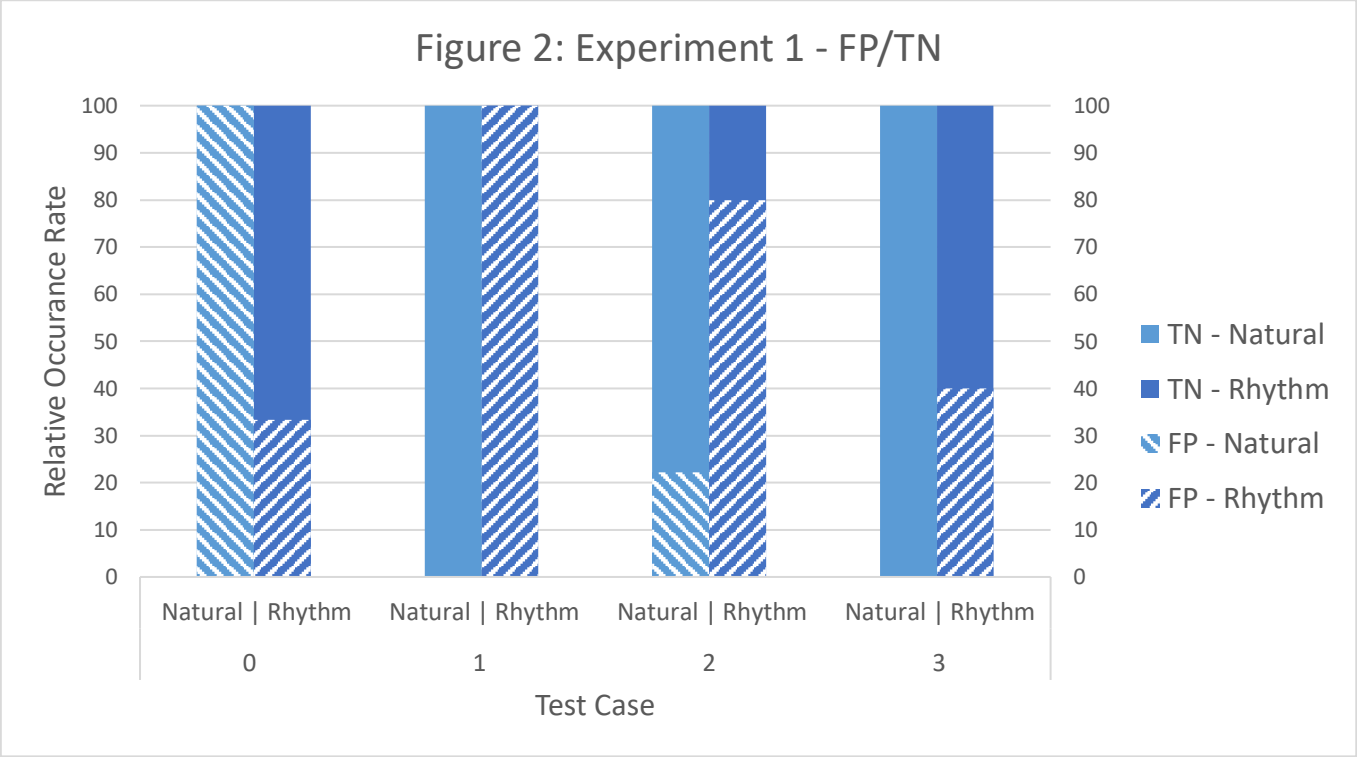
For volunteer 0, the use of rhythm was a hindrance – for 40% of their rhythmic attempts the volunteer was unable to reproduce the *ta-tum-ta-tum* pattern that they had chosen, having successfully reproduced their natural rhythm on every attempt.

For volunteer 1, rhythm had no impact, and they successfully authenticated on all attempts.

For volunteers 2 and 3, rhythm had the greatest possible impact, as they faithfully reproduced *The Can Can* and the *Mii Channel Music* respectively on every attempt, having been unable to match their natural rhythm on any attempt.

Across the four test cases there is a net benefit, although the results are quite inconsistent and could be improved with a greater sample size to determine if volunteers 2 and 3 really represent the norm, as well as how anomalous volunteers 0 really is.

To determine if using a known rhythm also makes it easier for a dishonest user to authenticate, I compared the proportions of FPs to TNs in much the same way, presented in Figure 2. For each test case attempts by the other volunteers were compared against the baselines from those assigned to each test case. For example, volunteers 1, 2, and 3 attempted to authenticate against volunteer 0's natural and chosen-rhythm baselines for test case 0.



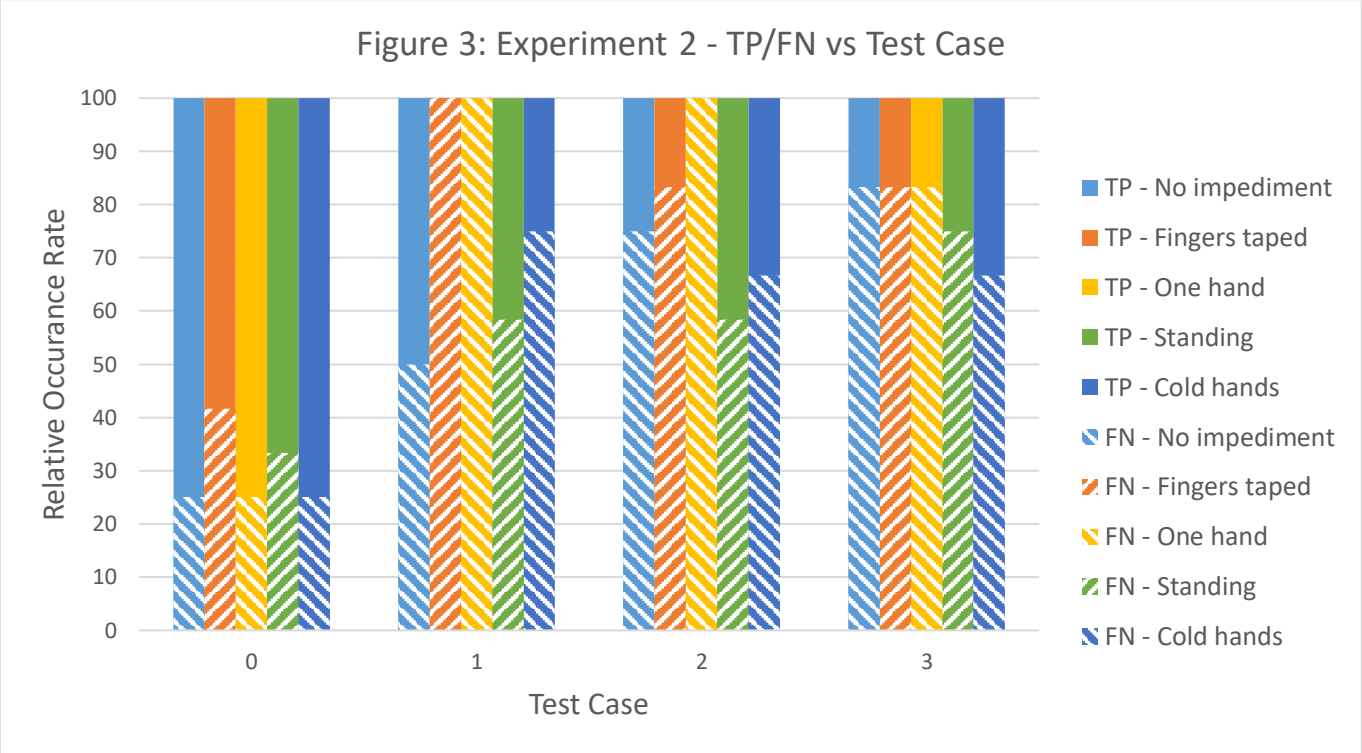
For test case 0, use of a rhythm significantly reduced the rate of FPs. This is likely due to “aaaaa” and “123456789” being so simple to type that all users naturally do so with similar cadence, and, whilst simple, *ta-tum-ta-tum* is perhaps not a rhythm the volunteers frequently encounter.

By contrast, whilst volunteer 1 had a sufficiently distinct typing cadence to prevent all entry with a natural rhythm, *Twinkle Twinkle Little Star* is so well-known that all attempts to match it were successful.

The same is true, although to a lesser extent, of *The Can Can* and *The Mii Channel Theme*, with the rhythmic FP rates higher than the natural FP rates for test cases 2 and 3.

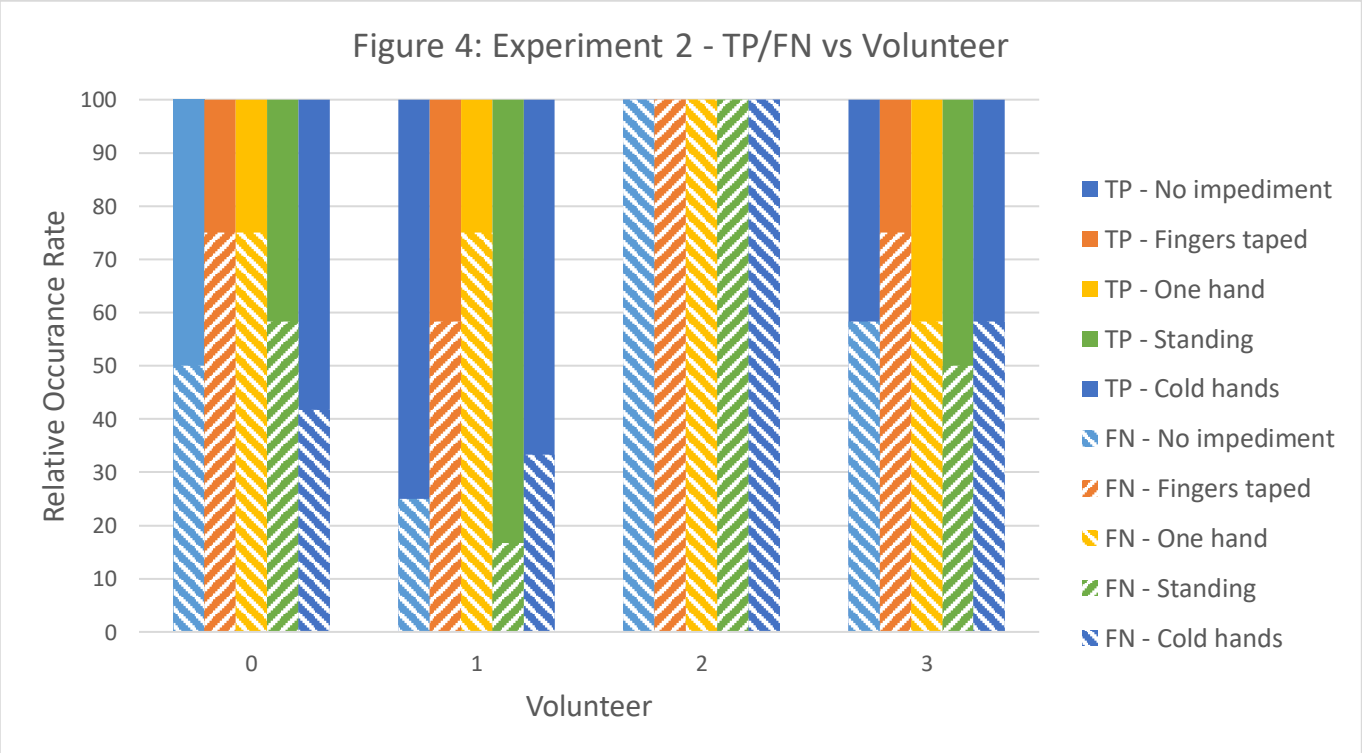
5.2. Experiment 2

To determine if the impediments prevent an honest user from authenticating, I compared the rate of TPs with no impediment against that of the impediments for each test case shown in Figure 3. Each user had three attempts per impediment (including no impediment) to authenticate against their own baseline.



If my hypothesis were correct, then the leftmost column of each group would show a lower rate of FN than the other columns in the group, but that does not appear to be the case – there is no readily observed trend at all, except to say that test case 0 was the easiest to authenticate against.

I then made the same comparison, but grouping by volunteer rather than test case, to discern if any of the volunteers showed the hypothesised behaviour, shown in Figure 4.



For the sake of analysing Figure 4 I will overlook volunteer 2, who clearly had some difficulty matching their own baselines. It is possible that they typed erratically when setting the baselines, making them difficult to match against. I believe it is worth noting that their 3 natural rhythm attempts all matched volunteer 0's baseline for test case 0 in Experiment 1, so volunteer 2 is capable of typing with some measure of consistency.

Figure 4 could support my hypothesis somewhat more than Figure 3: we can at least see that typing with taped fingers consistently increases the rate of FNs, and that for volunteers 0 and 1 so does typing with only one hand. A significant impact of typing one-handed for those volunteers was in struggling to type capital letters for test cases 2 and 3, as those volunteers used the shift key. However, volunteer 3 used caps lock, reducing the impact of only having one hand. Further investigation into the impact of typing style compounded with impediments was unfortunately beyond the scope of my experiments. Additionally, all volunteers were right-handed, and most of the characters in the test cases were on the left of the keyboard, which may have reduced the impact of only having one hand.

Typing whilst standing or with cold hands does not appear to have a consistent effect in my experimentation. The volunteers did note that holding the bag of frozen peas left them with cold palms but more or less comfortable fingers, so in retrospect it is not surprising that the effect was minimal. Anecdotally I was aware that standing can make it more difficult to touch-type, but given the volunteers were unfamiliar with both the keyboard and the <ID, PW> pairs, they were often not touch-typing whilst sat down, reducing the impact of being made to stand.

6. Further Work

I only had the capacity for alpha tests against two hypotheses, but my results could be improved upon and supplemented with further experimentation. An increased sample size would increase the credibility of my results, as well as help identification of outliers. Aside from simply repeating my experiments with a larger pool of volunteers and test cases, the areas I would consider most interesting to investigate are as follows:

- Allowing volunteers to become more familiar with <ID, PW> pairs before creating the baselines, as they would be with any passwords they were to actually use on a regular basis. I theorise that this would allow smaller margins of error without reducing the number of FNs, which would reduce the proportion of FPs (i.e. improve the security provided by the authentication).
- Testing the impact of unfamiliar hardware on FNs for known <ID, PW> pairs – does partially remote working become less practical if you need the same keyboard at home and at the office?
- Improved investigation of impediments: if the error margins could be reduced without eliminating unimpeded TPs, then perhaps there would be a more visible impact on the impeded attempts to authenticate.
- The impact of typing style on keystroke dynamic-based authentication: with percentage error margins, those who type slowly could see fewer FNs, or fast typing could correlate to experience typing and a more consistent cadence. Additionally, people who use fewer fingers when typing could be less impeded by reduced hand mobility from cold or injury.
- Expand the keylogger program and analysis to include misspelled IDs and passwords (as well as use of backspace and arrow keys to correct them before they are submitted): anecdotally, if I misspell a password on my first attempt I retry more slowly, which would not work for basic keystroke dynamic-based authentication.

7. Conclusion

I investigated if 1) Both TPs and FPs will be more common when users (attempt to) conform to a specific rhythm, and 2) The rate of FNs will increase (and TPs correspondingly decrease) when users are subject to minor impediments. Figure 1 showed that for most volunteers use of a rhythm did increase the rate of TPs, and Figure 2 showed that for most test cases use of a rhythm also increased the rate of FPs. This supports my first hypothesis, and suggests that rhythm-based authentication is a double-edged sword: those seeking to use it should choose a rhythm they know well but that is not well-known to the average person. Figures 3 and 4 did little to support my second hypothesis, but nor did they support the opposite of it (i.e. that minor impediments would increase the rate of TPs). This would suggest that keystroke dynamic-based authentication could be practical to use without fear of a cold morning locking a user out of a system, but I would prefer to undertake some of the further investigation discussed in Section 6 before saying that definitively.

8. References

[1] R. A. Maxion and K. S. Killourhy, “Keystroke biometrics with number-pad input,” in *2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, Chicago, 2010.

[2] J. Ilonen, “Keystroke dynamics,” in *Advanced Topics in Information Processing–Lecture*, Lappeenranta, 2003.

[3] A. O. Adesina and O. Oyebola, “An Investigation on the Impact of Age Group and Gender on the Authentication Performance of Keystroke Dynamics.,” *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 10, no. 9, pp. 18-21, 2021.

[4] N. Çevik, S. Akleylek and K. Yunus Koç, “Keystroke Dynamics Based Authentication System.,” in *2021 6th International Conference on Computer Science and Engineering (UBMK)*, 2021.

Appendix A: Code

```
...

echo -n "$volunteer,1,$test_case,$test_code" >>$output_file

...

function readID {
    # read ID
    while read -n1 key; do
        # if key = Escape key
        if [ $key == $'\e' ]; then
            break
        fi

        id=$id$key
        id_millis[$id_index]=$(date +%s%3N)

        ((id_index += 1))
        if [ $id_index -gt 9 ]; then
            break
        fi
    done

    echo ""

    # add ID and keystroke dynamics to CSV
    echo -n ",$id" >>$output_file
    for ((i = 0; i < $id_index - 1; i++)); do
        delay=$((${id_millis[i + 1]} - ${id_millis[i]}))
        echo -n ",$delay" >>$output_file
    done
    for ((i = $id_index; i < 10; i++)); do
        echo -n "," >>$output_file
    done
}

...

function readPW {
    while read -n1 key; do
        # if key = Escape key
        if [ $key == $'\e' ]; then
            break
        fi

        pw=$pw$key
        pw_millis[$pw_index]=$(date +%s%3N)
```

```
        ((pw_index += 1))
        if [ $pw_index -gt 19 ]; then
            break
        fi
    done

    echo ""

    echo -n ",$pw" >>$output_file
    for ((i = 0; i < $pw_index - 1; i++)); do
        delay=$((${pw_millis[i + 1]} - ${pw_millis[i]}))
        echo -n ",$delay" >>$output_file
    done
    for ((i = $pw_index; i < 19; i++)); do
        echo -n "," >>$output_file
    done

}

...

echo "" >>$output_file  # end of line
```

Appendix B: Experimental Data

Natural rhythm baselines extract

Volunteer	Test Case	ID	I1	I2	I3	I4	I5	I6	I7	I8	I9	Password	P1	P2	P3	P4	P5	P6
0	0	aaaaa	239	242	239	225	#N/A	#N/A	#N/A	#N/A	#N/A	123456789	218	208	239	240	264	241
0	1	edward	239	197	102	118	227	#N/A	#N/A	#N/A	#N/A	wedfvbghyu	66	328	71	424	66	238
0	2	jc311	107	333	208	153	#N/A	#N/A	#N/A	#N/A	#N/A	Password1	209	176	160	201	145	105
0	3	MrBig	207	280	209	107	#N/A	#N/A	#N/A	#N/A	#N/A	s3cr3t	235	249	246	93	137	#N/A
1	0	aaaaa	352	289	344	352	#N/A	#N/A	#N/A	#N/A	#N/A	123456789	248	288	283	328	280	328
1	1	edward	206	130	191	120	215	#N/A	#N/A	#N/A	#N/A	wedfvbghyu	199	298	176	328	142	361
1	2	jc311	360	392	247	177	#N/A	#N/A	#N/A	#N/A	#N/A	Password1	376	262	145	201	302	290
1	3	MrBig	360	424	312	272	#N/A	#N/A	#N/A	#N/A	#N/A	s3cr3t	264	408	272	288	264	#N/A
2	1	aaaaa	200	199	240	167	208	#N/A	#N/A	#N/A	#N/A	123456789	95	392	95	440	112	359
2	2	edward	111	289	240	168	#N/A	#N/A	#N/A	#N/A	#N/A	wedfvbghyu	239	248	136	247	104	200
2	3	jc311	272	329	247	128	#N/A	#N/A	#N/A	#N/A	#N/A	Password1	120	257	352	225	392	#N/A
2	0	MrBig	215	226	223	217	#N/A	#N/A	#N/A	#N/A	#N/A	s3cr3t	223	263	257	256	256	248
3	0	aaaaa	216	207	224	215	#N/A	#N/A	#N/A	#N/A	#N/A	123456789	233	240	247	272	295	304
...																		

Chosen rhythm upper bounds extract

Volunteer	Test Case	ID	I1	I2	I3	I4	I5	I6	I7	I8	I9	Password	P1	P2	P3	P4	P5
0	0	aaaaa	253.5	528	240	612	#N/A	#N/A	#N/A	#N/A	#N/A	123456789	85.5	528	84	528	72
1	1	edward	660	660	648	720	732	#N/A	#N/A	#N/A	#N/A	wedfvbghyu	648	672	708	720	708
2	2	jc311	588	600	312	315	#N/A	#N/A	#N/A	#N/A	#N/A	Password1	528	553.5	252	324	300
3	3	MrBig	1297.5	1486.5	927	840	#N/A	#N/A	#N/A	#N/A	#N/A	s3cr3t	792	552	696	766.5	636