Use Your Words: Designing One-time Pairing Codes to Improve User Experience

Sarah Wiseman
Goldsmiths,
University Of London
London, United Kingdom
s.wiseman@gold.ac.uk

Gustavo Soto Mino,
Anna L. Cox,
Sandy J.J. Gould
University College London
London, United Kingdom
{*gustavo.mino*.14|*anna.cox*|*sandy.gould*}
@ucl.ac.uk

Joanne Moore,
Chris Needham
BBC
London, United Kingdom
{*joanne.moore*|*chris.needham*}
@bbc.co.uk

ABSTRACT
The Internet of Things is connecting an ever-increasing number of devices. These devices often require access to personal information, but their meagre user interfaces usually do not permit traditional modes of authentication. On such devices, one-time pairing codes are often used instead. This pairing process can involve transcribing randomly generated alphanumeric codes, which can be frustrating, slow and error-prone. In this paper, we present an improved pairing method that uses sets of English words instead of random strings. The word method, although longer in terms of character length, allows users to pair devices more quickly, whilst still maintaining the complexity necessary for secure interactions.

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

Author Keywords
Pairing; passcodes; passwords; data entry; authentication; Internet Of Things; typing; transcription; text entry

INTRODUCTION
Thanks to innovations in technology and the Internet of Things (IoT), it is becoming more common for previously "dumb" devices to become smart. Everyday interactions with lights [8], heating [18] or even washing machines [4] can be conducted with sophisticated remote controls that are accessed via smartphones. However, linking these devices to the associated web accounts can be difficult if they do not come pre-paired. Providing your email address, for example, to a smart TV using only a standard remote control can be tedious.

The alternative to this is to allow a 'pairing' process. In this scenario, the device displays a unique code, which a user must then enter into a web-based account on a laptop, smart phone, tablet or any device capable of browsing the web. The IoT device is then linked with the account, and all future interaction can take place online, using a device that is better designed for complex user interaction.

This is a pre-print.
Published version (see DOI, below)
to appear at ACM CHI 2016

The idea of generating a code in order to create a pairing of two devices is not new [11] and has been used commonly to allow people to, for example, remotely access secure systems when working from home [9]. The current standard in these situations is to generate a numeric or alphanumeric string for the user to transcribe. However, these strings can be awkward to type on touchscreen devices [10] and difficult to memorise [5], forcing multiple glances between the input screen and the device displaying the code.

When trying to improve the design of systems that require data transcription, researchers have focused on redesigning the process [6, 26] or the data entry interface [7, 27]. Rarely is there an opportunity to design the data string itself. In the context of a pairing action, codes are not semantically meaningful and it is therefore possible to design pairing codes that are optimized for their use in practice. In this way, we can explore the design not only of the underlying technology, but the data they use as well.

We present an investigation of alternatives to standard alphanumeric string codes. We show that by using a set of short words, the pairing process can be made faster and less taxing for users. This method not only improves the experience for the user, it also offers a sufficient number of permutations to generate multiple, non-clashing codes. This allows many users to pair at the same time and reduces the chances of security breaches due to malicious attacks.

RELATED LITERATURE
Pairing processes are used to connect devices so that they can share information. It is possible to complete this process through technology alone, for instance using an infrared channel [2]. However, this requires that both devices be in close proximity to one another, and that both have the specialized technology necessary for physical communication. An alternative to this method is to ask the user to verify and authenticate the device. There are a number of possible processes that can be utilized, from asking users to compare two values on two different devices, to selecting the correct value from a

DOI: http://dx.doi.org/10.1145/2858036.2858377

possible list, to copying a value from one device into another [25]. Research has shown that, although Selection is preferred by users, the Copying method is the most secure [25]. Indeed, many systems use this method to display a text-based code to a user in order to verify a pairing. An example of this process is seen on banking websites, which display numeric codes which must be given to a phone operator to verify the caller's identity. Although text-based codes are common, alternative pairing solutions have been suggested that make use of pictorial comparisons [20] or comparing sounds [21]. However, text-based numeric or alphanumeric passcodes have been found to be safer [15, 17] and users show strong preference for word-based comparisons [16]. In addition, sound- or picture-based systems require that the display device has the ability to display images and create sounds; something many IoT devices with simple displays cannot do.

The design of the text-based passcode can be compared to the generation of passwords. A password, unlike a passcode however, is used multiple times and must be memorized by the user. When a user generates a password, they are often required to create a complex string containing special characters to ensure a high level of security. Research shows that this can result in users forgetting their passwords or using workarounds [1, 14].

It is not just recall that can cause issues with complex string passwords, but the process of entry can itself be problematic. Research into the manual processes associated with entering complex passwords also suggests that the use of alphanumeric passwords can be problematic for smartphone users [10]. This stems largely from the need to switch between multiple different keyboard levels in order to navigate to the various characters required. However, passcodes often contain multiple character types, as this ensures a high level of complexity and thus better security.

SCENARIO

This work was completed in response to a request from a large broadcasting company's requirement for pairing codes to enable users to pair smart devices (such as radios and TVs) with their personal accounts [19]. This would allow users to save songs or programmes from their devices to their account for later use. The issue faced in this scenario is that the entry of account details is difficult on a TV remote, and is near impossible on a radio. Additionally, the output of a radio is particularly limited, often using only a 16 character LCD display to scroll song information. The solution to this problem is to issue temporary pairing codes that can be displayed on the smart device and then entered on a separate device better designed for input, such as a laptop or smartphone. Once logged-in to the personal account on a browser-capable device, the generated code can be entered, joining that account to the smart device so that all future interactions can be linked to the account.

The important factor here is that the codes need to be easily read and entered by users, and additionally that the number of possible codes needs to be large enough to ensure that during moments of high levels of pairing, there are enough possible codes to issue to users. For instance, a suggestion to pair on a TV programme which would result in large numbers of people pairing their devices at the same time. Additionally, there are security constraints to prevent brute force attacks from linking a malicious device to the user's account.

The requirements in this particular scenario called for a code set in the magnitude of 500 million. Commonly, text-based pass codes use either numeric or alphanumeric randomly generated strings. The aim of our investigation was to improve the readability, memorability and typability of the codes. Familiar text is faster to type than randomised strings (see point 5 in [22] for a summary), suggesting that word-based codes may perform better than randomly generated numeric or alphanumeric codes. We conducted an experiment to test this hypothesis.

METHOD

Participants

The experiment was conducted with 20 participants (13 female) with a mean age 27 (*SD*=5). The participants were recruited from a university subject pool and social media; they were reimbursed £8 for the 35 minute session.

Design

The experiment used a within-subjects design, with each participant taking part in each of the six conditions. These conditions were presented in two different orders to the participants, counterbalanced for device.

The 2x3 design used two factors: device and code type. Two devices were used in the experiment: a laptop and a touchscreen smart phone. The three levels of code type were numeric, alphanumeric and words. The dependent variables include time per trial, number of uncorrected errors, eye glances to the display device and NASA TLX scores which allow the participants to self-assess their workload during the task [12].

Materials

Table 1 shows the complexity of the three types of codes used in this experiment: numeric, alphanumeric and word-based. Note that each offers more than 500 million possible codes.

The numeric codes were generated as a random string of nine digits from 0-9, displayed in three blocks of three digits. The alphanumeric codes were similarly randomly generated, using the characters a-z, A-Z and 0-9, removing characters that could be easily visually confused (i, o, l, I, O ,L). From this set, a string of five characters was created and chunked into two blocks of two then three characters as splitting groups of alphanumeric characters can make them more memorable, which aids in the transcription process [23]. A word code was generated by randomly sampling three words from a set of English words and displaying them with spaces between each word. For the word codes to be feasible, the number of possible combinations of words would need to exceed 500 million in order to be comparable to the complexity of the numeric and alphanumeric codes. There are around 1100 three-letter English words, although many of these are likely to be obscure. When computing the word code complexity, a conservative estimate of 800 possible words was used, thus resulting in

Code Name	*Size*	*Example*	*Complexity*
Numeric	9 digits	180 487 101	10^9
Alphanumeric	5 characters	s4 5vB	56^5
Words	3 words	pot bee rid	800^3

Table 1. Different code generation schemes with set size. Numeric codes include digits 0-9. Alphanumeric codes include digits and all upper and lower case letters (excluding i,o,l,I,O,L). Word codes include three 3-letter English words.

$800^3 = 512,000,000$ possible combinations of three three-letter words. This estimate was based upon an upper limit of 1135 three-letter words found in the Unix "words" file and a lower limit of 754 three-letter words reported in [24], which took words from a spell-checking dictionary and thus excluded highly obscure words. For the purposes of the experiment, the word codes were generated from a list of 242 common English three-letter words taken from the easily accessible Google Books database.

The experiment was set up as two websites: one display and one input. The display website ran on a 22 inch computer monitor, whilst the input website ran on either a laptop (a MacBook Pro) or the participant's smartphone depending on the condition. A Tobii X120 eye-tracking device was set up to record the glances at the display screen.

The display screen showed the codes in 32px Helvetica font. The input screen displayed a text entry box and an enter button in the centre of the screen. In the laptop condition, the participant copied the code into this box using the physical keyboard. In the phone condition, the user used the installed keyboard on their smart phone to enter the code.

Procedure

Participants completed the experiment sat in front of the display monitor. Prior to two test trials, the eye tracker was calibrated for each participant. Each participant was able to freely move their head throughout the experiment. The 20 participants were counterbalanced: half entered codes on the laptop and then their smartphone whereas the the other half used the smart phone first and then the laptop.

The display screen began by showing a fixation cross for 3 seconds in the centre of the screen. After this, the first code was displayed in the centre of the screen. Once the user had entered the code on the separate input device, the fixation cross on the display screen would appear for 3 seconds before displaying the next code to be entered.

In each of the six conditions (two devices and three code types) the user copied 20 codes, for a total of 120 codes. The duration of code entry was measured from the point at which the code was shown on the display screen to the point the user pressed the 'enter' key on the input screen. After each condition, the participants were given a NASA TLX survey to fill in to self-assess the workload. This was completed six times in total.

After all trials had been completed, the participants were asked about their preference for the different codes; they were asked which code they found easiest to input on the laptop, then the smartphone. Finally, the participants were asked which code they would find easiest and safest to use in a device pairing situation.

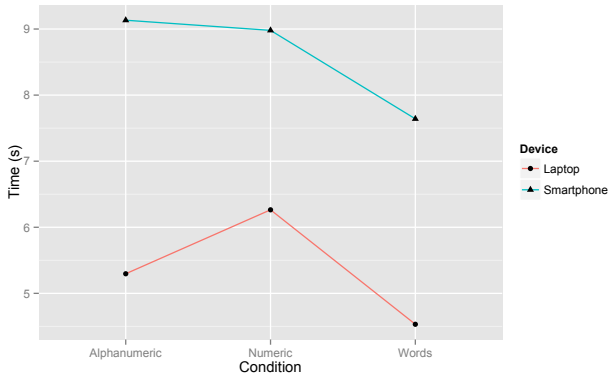

Figure 1. Mean time across all participants to enter one code

RESULTS

Speed of Transcription

Figure 1 shows the mean time taken to enter one code in each condition. Participants entered codes faster on the laptop (M=5.36-s, SD=1.97-s) than on the smartphone (M=8.58-s, SD=2.72-s). A two-way repeated ANOVA analysis of this result suggests that this difference was significant ($F(1,19)$=65.04 p<.001). Despite being longer in terms of character length, in both device conditions, the word codes were faster to enter (M=6.09-s, SD=2.24-s) than both the alphanumeric (M=7.62-s, SD=3.06-s) and numeric (M=7.21-s, SD=3.07-s) conditions. Analysis suggests that this speed difference is significant ($F(2,38)$=6.72 p=.003). A post-hoc pairwise comparison suggests that words are significantly faster to type than both numeric (p=.010) and alphanumeric codes (p<.001). The interaction between device and condition was non-significant, indicating that the advantage of word codes is not device-specific.

Error Rate

Error rate was calculated as the percentage of trials that resulted in the code being entered incorrectly. The mean error rate for codes entered in the laptop condition (M=3%, SD=4.6%) and smartphone condition (M=4.3%, SD=6.1%) was not significantly different. Similarly, the differences between error rates for numeric codes (M=3%, SD=4.9%), alphanumeric codes (M=4.3%, SD=5.9%) and word codes (M=3.8%, SD=5.5%) were also non-significant.

Eye gaze data

The eye tracking data was not recorded for six participants due to an equipment limitation. Additionally, the eye tracking data for the smartphone condition was unusable because of excessive head movement in this condition. In the laptop condition however, the data showed that the mean number of gazes to screen in the numeric condition was 1.8 (SD=0.73), in the alphanumeric condition it was 1.76 (SD=0.74) whereas in

the word condition the mean glances to the display screen was 1.3 (SD=0.53). Analysis suggests this difference is significant ($F(2,26)=3.53$ $p=.044$).

Workload analysis

The NASA TLX values were summed per condition for each participant (this produced values between 0 and 20 for each of the six conditions). The reported workload for entering the codes on the laptop was lower (M=37.37, SD=21) than the reported score for the smartphone (M=45.98, SD=22.45). Analysis suggests this difference is significant ($F(1,19)=19.86$ $p<.001$). The reported workload for entering word-based codes was lower (M=31.15, SD=17.22) than both numeric (M=44.95, SD=22.72) and alphanumeric (M=48.78, SD=22.33) codes. A repeated measures ANOVA test suggests this difference is significant ($F(2,38)=36.65$ $p<.001$). A post-hoc pairwise comparison suggests that word codes resulted in a significantly lower reported workload than both numeric ($p<.001$) and alphanumeric ($p<.001$) codes. The difference between workload for numeric and alphanumeric codes was non-significant, as was the interaction between device and condition.

Code Style Preference

When asked which code the participants preferred when using the laptop, 85% chose the word-based codes and 15% chose numeric codes. In the smartphone condition, 80% of participants preferred word-based codes, 15% preferred numeric codes and 5% preferred alphanumeric codes. In a device pairing scenario where security was emphasized as an important factor, 75% of users said they would prefer the word-based codes and 25% said they would prefer the numeric codes.

DISCUSSION

The alphanumeric condition required users to type five characters, whereas the numeric and word conditions required the users to type nine characters. Despite this difference in length, the words condition resulted in the fastest code entry time on both devices with no reduction in accuracy.

It is known that typing familiar words is faster than typing random strings [22]. The results of this experiment replicate this finding, as the randomly generated strings in the numeric and alphanumeric conditions were transcribed slower than the more familiar words. This cognitive facilitation of familiarity during the transcription process may explain the increase in speed for word-based codes. The eye gaze data suggests that the word-based codes were easier to store in short term memory, with participants making fewer glances back to the display screen in the word condition compared to the other two conditions.

The device used had a significant effect upon the speed of code entry also, with codes being entered significantly faster on the laptop than on the users' smartphones. This replicates research that has found typing on physical keyboards faster than touchscreens or flat keyboards, particularly on smaller mobile phone screens [13, 3].

Ultimately, in a pairing situation, the key measure is that the method used should be no more error-prone than an alternative. This suggests that any of the three conditions tested here would be usable. However, the indications of user preference and perceived workload alongside the improved speed suggest that the word-based codes are better than number or alphanumeric codes, whilst maintaining a similar level of complexity.

Further to this analysis, the words condition also has a "safer" fail mode than the other two code options. For example when entering an alphanumeric code, if the user was to slip whilst typing and enter 'rT 5v6' instead of 'rT 5v5', this incorrect code is still valid in the system, meaning there is a chance it could represent a code another user has been given and thus incorrectly pairing their device with the other user's account. In the word-based code condition, a user accidentally entering 'win cat trt' instead of 'win cat try' has entered a string of characters that does not constitute a word and therefore represents an invalid code. The user can then be immediately alerted to the error. Although it is still possible that a typing slip when using the words system could produce a valid code, the likelihood of this happening is greatly reduced compared to other systems. This means the word-based codes are less likely to result in accidental pairings of devices and accounts: the syntactical rules inherent in word creation mean that word-based codes have some level of built-in error checking.

Gallagher and Byrne, in [10], highlight the problems of entering mixed character passcodes on touchscreen devices such as a smartphone, where each switch between keyboard levels adds time to the text entry process. In the current study, participants were required to navigate multiple keyboard levels in the alphanumeric condition when using the smartphone (switching keyboard between lower case letters, upper case letters and numbers). Despite the alphanumeric codes being 45% the length of the numeric codes, the time difference between these two conditions was not significant. This finding appears to support the previous research that suggests switching keyboard levels increases time for data input.

CONCLUSION

The results of this study strongly support the use of word-based codes in a device pairing context. Not only are word-based codes faster to transcribe, they also require fewer glances to the display screen to memorise, result in a lower perceived workload and are preferred by users. These improvements in the pairing process are not associated with an increase in error rate, nor do they result in codes that are less secure: the search space of possible codes is still in the same order of magnitude as numeric codes of the same length, and is greater than the needs of an example scenario set by a company using a pairing system. This research suggests that when pairing devices with poor input interfaces, and low fidelity output displays, word-based codes are a preferable alternative to the current standard of randomly generated numeric and alphanumeric codes.

ACKNOWLEDGMENTS

This work was supported by the UK Engineering and Physical Sciences Research Council grant EP/L504889/1. We thank the anonymous reviewers of this manuscript for their insight.

REFERENCES

1. Anne Adams and Martina Angela Sasse. 1999. Users are not the enemy. *Commun. ACM* 42, 12 (1999), 40–46. DOI:http://dx.doi.org/10.1145/322796.322806
2. Dirk Balfanz, Diana K Smetters, Paul Stewart, and H Chi Wong. 2002. Talking to Strangers: Authentication in Ad-Hoc Wireless Networks. In *NDSS*.
3. Julia Barret and Helmut Krueger. 1994. Performance effects of reduced proprioceptive feedback on touch typists and casual users in a typing task. *Behaviour & Information Technology* 13, 6 (1994). DOI: http://dx.doi.org/10.1080/01449299408914618
4. Berg Cloud Limited. 2014. Cloud Wash. (2014). http://bergcloud.com/case-studies/cloudwash/
5. Roy Brener. 1940. An experimental investigation of memory span. *Journal of Experimental Psychology* 26, 5 (1940), 467–482. DOI: http://dx.doi.org/10.1037/h0061096
6. Duncan P. Brumby, Anna L. Cox, Jonathan Back, and Sandy J. J. Gould. 2013. Recovering from an interruption: Investigating speed-accuracy trade-offs in task resumption behavior. *Journal of Experimental Psychology: Applied* 19, 2 (2013), 95–107. DOI: http://dx.doi.org/10.1037/a0032696
7. Abigail Cauchi, Andy Gimblett, Harold Thimbleby, Paul Curzon, and Paolo Masci. 2012. Safer "5-key" number entry user interfaces using Differential Formal Analysis. In *Proceedings of the 2012 BCS Conference on Human-Computer Interaction*. 29–38.
8. Easy Bulb. 2015. Easy Bulb. (2015). http://easybulb.com/
9. EMC Corporation. 2015. RSA. (2015). http://www.emc.com/domains/rsa/index.htm
10. Melissa A Gallagher and Michael D Byrne. 2015. Modeling Password Entry on a Mobile Device Modeling. In *Proceedings of the International Conference on Cognitive Modeling*. 45–50.
11. Richard H Guski, Raymond C Larson, Syephen M Matyas Jr, Donald B Johnson, and Don Coppersmith. 1997. Authentication system using one-time passwords. (1997). https://www.google.com/patents/US5592553
12. Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Advances in psychology* 52 (1988), 139–183. DOI: http://dx.doi.org/10.1016/S0166-4115(08)62386-9
13. Eve Hoggan, Stephen A Brewster, and Jody Johnston. 2008. Investigating the Effectiveness of Tactile Feedback for Mobile Touchscreens. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1573–1582. DOI: http://dx.doi.org/10.1145/1357054.1357300
14. Phillip Inglesant and M. Angela Sasse. 2010. The true cost of unusable password policies: password use in the wild. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 383–392. DOI: http://dx.doi.org/10.1145/1753326.1753384
15. Ronald Kainda, Ivan Flechais, and A.W. Roscoe. 2009. Usability and Security of Out-Of-Band Channels in Secure Device Pairing Protocols. In *Symposium on Usable Privacy and Security*. DOI: http://dx.doi.org/10.1145/1572532.1572547
16. Ronald Kainda, Ivan Flechais, and A. W. Roscoe. 2010. Secure and Usable Out-Of-Band Channels for Ad Hoc Mobile Device Interactions. *Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices* 6033 (2010), 308–315. DOI: http://dx.doi.org/10.1007/978-3-642-12368-9
17. Alfred Kobsa, Gene Tsudik, and Yang Wang. 2009. Serial Hook-ups : A Comparative Usability Study of Secure Device Pairing Methods. In *Proceedings of the 5th Symposium on Usable Privacy and Security*. DOI: http://dx.doi.org/10.1145/1572532.1572546
18. Nest Labs. 2015. Nest. (2015). https://nest.com/
19. Sean O'Haplin, Chris Needham, and Michael Barroco. 2014. *The Cross Platform Authentication Protocol*. Technical Report September. EBU Operating Eurovision And Euroradio, Geneva. 1–39 pages.
20. Adrian Perrig and Dawn Song. 1999. Hash Visualization : a New Technique to improve Real-World Security. *International Workshop on Cryptographic Techniques and E-Commerce* (1999), 1–8.
21. Ramnath Prasad and Nitesh Saxena. 2008. Efficient device pairing using "human-comparable" synchronized audiovisual patterns. In *Applied Cryptography and Network Security*. 328–345. DOI: http://dx.doi.org/10.1007/978-3-540-68914-0{_}20
22. Timothy A. Salthouse. 1986. Perceptual, cognitive, and motoric aspects of transcription typing. *Psychological bulletin* 99, 3 (1986), 303–319. DOI: http://dx.doi.org/10.1037/0033-2909.99.3.303
23. Jan Maarten Schraagen and Kees van Dongen. 2005. Designing a licence plate for memorability. *Ergonomics* 48, 7 (2005), 796–806. DOI: http://dx.doi.org/10.1080/00140130500123720
24. Reginald Smith. 2012. Distinct word length frequencies: distributions and symbol entropies. *Glottometrics* 23 (2012), 7–22.
25. Ersin Uzun, Kristiina Karvonen, and N. Asokan. 2007. Usability Analysis of Secure Pairing Methods. *Financial Cryptography and Data Security* (2007), 307–324. DOI: http://dx.doi.org/10.1007/978-3-540-77366-5_29
26. Sarah Wiseman, Anna L. Cox, Duncan P. Brumby, Sandy J. J. Gould, and Sarah O'Carroll. 2013a. Using Checksums to Detect Number Entry Error. In

Proceedings of the SIGCHI Conference on Human Factors in Computing. 2403–2406. DOI: http://dx.doi.org/10.1145/2470654.2481332

27. Sarah Wiseman, Orla Hennessy, Anna L. Cox, and Duncan P. Brumby. 2013b. Tailoring Number Entry Interfaces To The Task of Programming Medical Infusion Pumps. In *International Annual Meeting of the Human Factors and Ergonomics Society*. 683–687. DOI: http://dx.doi.org/10.1177/1541931213571148