Keystroke dynamics

Jarmo Ilonen[1]

Lappeenranta University of Technology, Skinnarilankatu 34, 53850 Lappeenranta, Finland,
ilonen@lut.fi

Abstract. This article is an introduction to keystroke dynamics. Keystroke dynamics is a biometric which is based on assumption that people type in uniquely characteristic manners. Keystroke dynamics is mainly used for verification, but also identification is possible. Also commercial products exist. Additionally, keystroke dynamics can be used to eavesdrop secure communications by guessing what was written based on timings between letters.

1 Introduction

Keystroke dynamics is a biometric based on assumption that different people type in uniquely characteristic manners. Observation of telegraph operators in the 19th century revealed personally distinctive patterns when keying messages over telegraph lines, and telegraph operators could recognize each other based on only their keying dynamics. Conceptually closest correspondence among biometric identification systems is signature recognition. In both signature recognition and keystroke dynamics the person is identified by their writing dynamics which are assumed to be unique to a large degree among different people. Keystroke dynamics is known with a few different names: keyboard dynamics, keystroke analysis, typing biometrics and typing rhythms. [1, 2]

Access to computer systems is usually controlled by user accounts with usernames and passwords. Such scheme has little security if the information falls to wrong hands. Key cards or biometric systems, for example fingerprints, can be used to strengthen security, but they require quite expensive additional hardware. On the other hand keystroke dynamics can be used without any additional hardware. Also, the user acceptance of a keystroke dynamics biometric system is very high, since users do not necessarily even notice that such system is used.

Keystroke dynamics is mostly applicable to verification, but also identification is possible. In verification it is known who the user is supposed to be and the biometric system should verify if the user is who he claims to be. In identification, the biometric system should identify the user without any additional knowledge, using only keystroke dynamics. Most applications of keystroke dynamics are in field of verification.

Specifics of keystroke dynamics and features used with keystroke dynamics are presented in Section 2. Keystroke dynamics methods and applications used

for verifying and identifying users are presented in Section 3. Darker side of computer security – cracking passwords and eavesdropping secure communications – has also its uses for keystroke dynamics. By detecting timings of keystrokes from a secure communication channel and knowing keystroke dynamics profile of the user it may be possible to sniff passwords or other text that the user writes. This topic is presented with an example in Section 4.

2 Features used with keystroke dynamics

Keystroke dynamics include several different measurements which can be detected when the user presses keys in the keyboard. Possible measurements include:

- Latency between consecutive keystrokes.
- Duration of the keystroke, hold-time.
- Overall typing speed.
- Frequency of errors (how often the user has to use backspace).
- The habit of using additional keys in the keyboard, for example writing numbers with the numpad.
- In what order does the user press keys when writing capital letters, is shift or the letter key released first.
- The force used when hitting keys while typing (requires a special keyboard).

Statistics can be either global, i.e, combined for all keys, or they can be gathered for every key or keystroke separately. Systems do not necessarily employ all of these features. Most of the applications measure only latencies between consecutive keystrokes or durations of keystrokes. In Figure 1 is an example of writing word "password" several times and measuring latencies between keystrokes. Timings have been measured for three different persons. There are clear differences in latencies and their standard deviations.

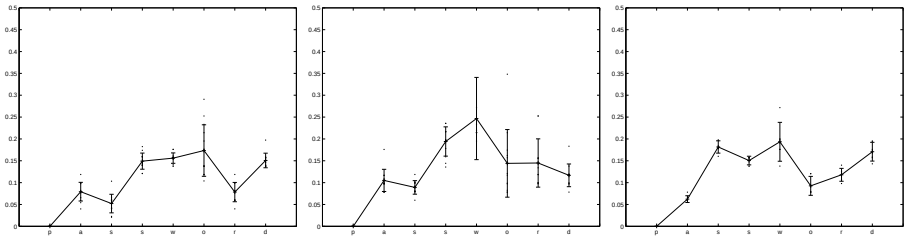


Fig. 1. Latencies between keystrokes when writing word "password" by three different persons. The word was written several times. The lines represent average latencies, errorbars represent standard deviations.

Latencies between keystrokes and durations of keystrokes are popular measurements because they can be easily measured with normal PC hardware. Both

key *press* and *release* events generate hardware interrupts. Gathering keystroke dynamics data has, however, few complications. Several keys can be pressed at the same time – the user presses the next key before releasing the previous one – and it happens quite often when writing fast. Depending on what is measured, there might even be negative time between releasing a key and pressing the next. It also adds slightly to complexity of the keystroke dynamics system if it is wanted to know when the user presses SHIFT, ALT and other special keys. [3]

Another challenge is that there is a very wide variety of typing skills, and the biometric systems should work for all users. First of all, the speed of typing can be wildly different between diffent users. An experienced touch-typist writes easily several tens of times faster than a beginner using "hunt-and-peck" style with one finger. Also the predictability of a fast writer is much greater – there is no need to stop and think where some letter is located on the keyboard. The typing can also be affected if the user is on a lower level of alertness, for example sleepy or ill. Users will additionally sometimes have accidents and consequently write in an ab-normal fashion for a few weeks when a finger is bandaged, or type with one hand when holding a coffee cup in other hand, and so on. Changing keyboard to a different model or using a laptop computer instead of a normal PC can also affect keystroke dynamics tremendously. All these factors have to be taken into account when designing a keystroke dynamics system.

3 User verification and identification

Keystroke dynamics systems can be used for both verification and identification. They have clearly different applications:

1. Verification: Identity of the user is verified usually at log-in time by measuring the typing pattern when writing the username and the password and comparing measurements to a previously stored profile.
2. Identification: A larger amount of keystroke dynamics data is collected, and the user of the computer is identified based on previously collected information of keystroke dynamics profiles of all users.

Verification has more directly applicable uses with keyboard dynamics than identification and it is a far more studied subject.

Goodness of a biometric system is determined by type I and type II error rates. Type I error means rejection of a valid user and type II error means an acceptance of an invalid user. In the first case a valid user gets annoyed because he could not log into the system. In the second case a user without proper authorization (an attacker) could log into the system. Both error rates should be optimally 0%. There is a balance between the two error types. If one kind of error is made lower by tuning parameters of a method, the other usually gets higher. For example, if all users are accepted – the biometric authorization is basically turned of – type I error will be 0% but type II error will be very high since every user is accepted. From a security point of view type II errors should be minimized – no chance for an illegal user to log in. However, type I errors

should also be infrequent because valid users get annoyed if the system makes their lives harder.

Both verification and identification methods are presented in following sections together with few examples from previous studies. A brief introduction to a commercial system, BioPassword, is also included.

3.1 Verification

Access to computer systems in mainly controlled by user accounts: usernames and passwords. If someone knows a username together with the password, one can access the computer system. Passwords are often quite easy to guess. They may have some direct connection to the person the account belongs to (birthday, name of a family member or pet, and so on), they may be normal dictionary words which are easily guessed by trying all of them, or the password might actually be written on a post-it note attached to somewhere near the computer. So, there is a clear need for strengthening the password based authentication. Keystroke dynamics is a sensible choice because a normal username/password scheme can be easily extended to use also keystroke dynamics and there is no need for additional hardware. [1]

Using keystroke dynamics as an addition to normal password based authentication is quite straighforward. When the password is changed (or created the first time), following steps are followed:

1. The new password is written by the user several times.
2. A profile of keystroke dynamics is created, for example by measuring latencies between consecutive keypresses and calculating their averages and standard deviations.
3. The profile is stored as an addition to the encrypted password.

The user has to write anyway the new password several times, usually twice, when changing it, so the password changing procedure is not changed much. Two times might not be enough for creating a usable profile, but a few more repetitions might be needed. Example of a profile was presented earlier in Figure 1. There are no user-visible changes in the authentication procedure, even though internally the procedure has a few more steps which are illustrated in Figure 2. Basically keystroke dynamics are measured when the password is written and measurements are compared to the formerly created profile. The system can additionally do similar checks when writing the username.

There have been a lot of studies on using keystroke dynamics for user verification [1, 4–7]. Most studies have used durations between keystrokes as features for user verification, but some have also used keystroke durations (the time a key is held down). All studies use two stages: 1) learning users keystroke dynamics (enrollment), and 2) comparing new data to the profile collected in stage 1. Stage 1 consists of writing the username and password several times, though sometimes only usernames are used, and forming a profile. The type of profile depends on the used classification method. Used classification methods include traditional statistic techniques, Bayesian classifiers, neural networks and fuzzy systems.

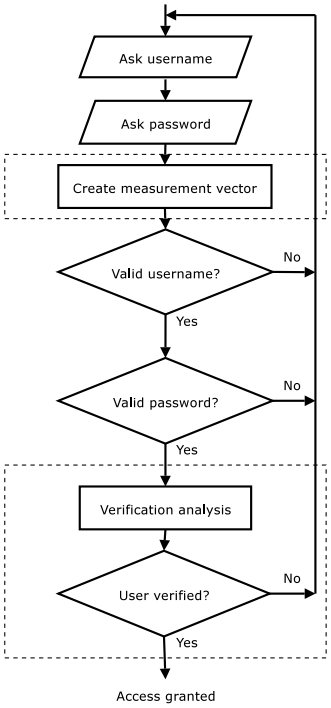


Fig. 2. Authentication. Dotted areas are added to the normal authentication procedure.

"Computer-access security systems using keystroke dynamics" by S. Bleha et al. [4] is one of the most referenced studies on keystroke dynamics is. In this study users only had a username, there was no separate password. The validity of the user was decided solely based on how they wrote their usernames, kind of using the username as the signature of the user. Latencies between keystrokes were used as features. Thirty latest valid username entries were used as reference pattern when deciding if the user is valid. Two different classification methods were used: minimum distance classifier and Bayesian classifier.

The normalized minimum distance classifier used for verification was

$$D_i = \frac{(X - m_i)^t (X - m_i)}{\|X\| \|m_i\|}. \tag{1}$$

And the normalized Bayesian classifier was

$$d_i = \frac{(X - m_i)^t C_i^{-1} (X - m_i)}{\|X\| \|m_i\|}. \tag{2}$$

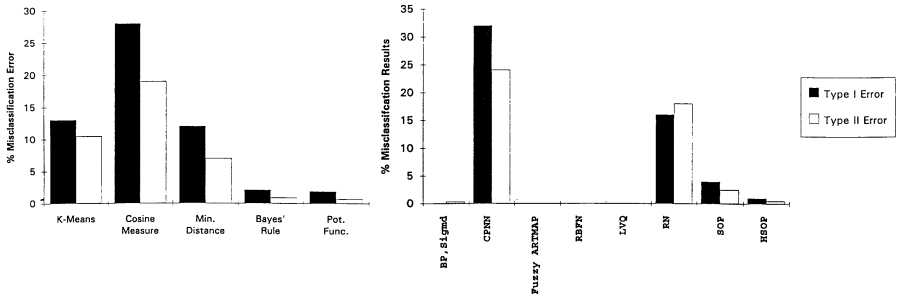
In both equations the participant is claiming to be user i, X is the test vector (latencies between keystrokes), m_i is the average of previous attempts for the

user and C_i is the covariance matrix. The normalization is done to accommodate for differences in password lengths. There is a defined threshold for deciding whether the user is accepted or not. Thresholds are different for both classifiers. Threshold of 0.030 was used for the minimum distance classifier and 0.000030 for the Bayesian classifier. If the user was not accepted with the first trial the second trial was given with reduced thresholds, 0.0029 and 0.000029 respectively. Both classifiers are used together to decide if the user is valid. The user is rejected only when both classifier thresholds are exceeded. In the experiments there were 10 valid users and 22 individuals tested the system as invalid users. The invalid users had a chance to observe valid users so they could try to imitate their writing styles. The results are presented in Table 1. 23 out of the 44 rejections of valid users were caused by two participants who were not used to PC keyboards, and did not use them other than during tests. 15 out of the 22 acceptances of invalid users were for a slow two-finger typist whose writing style was easy for others to imitate.

Table 1. Results for user verification. [4]

	Rejection of valid users (Type I error)	Acceptance of invalid users (Type II error)
Total attempts	539	768
Errors	44	22
% error	8.1%	2.8%

"Verification of computer users using keystroke dynamics" by M. S. Obaidat and B. Sadoun [1] is a comprehensive study of different classification methods that can be used with keystroke dynamics. Both latencies between keystrokes and keystroke durations were used as features with numerous different statistical and neural classification methods. It was noted that keystrore durations gave better results than latencies between keystrokes, but using both measurements together gave the best results. There were 15 valid users and 15 invalid users who tried to get access to all of valid accounts. Results for different methods are presented in Figure 3 when both keystroke durations and latencies between keystrokes have been used. Best results, 0% of both type I and type II errors, were achieved with following neural methods: Fuzzy ARTMAP (a generalization of adaptive resonance theory networks (ART) with fuzzy set theory operations), RBFN (Radial Basis Function Network) and LVQ (Learning Vector Quantization). Best results with statistical methods were achieved with the potential function and the Bayes rule, though many of the neural methods gave better results.



(a) (b)

Fig. 3. User verification results for (a) statistical and (b) neural methods when both latencies between keystrokes and key hold-times have been used. Note that the scales are different. [1]

3.2 Identification

In case of keystroke dynamics identification means that the user has to be identified without additional information besides measuring his keystroke dynamics. A short predefined text could be used for identification, i.e., profiles of all the users typing a certain text are stored and later the user is identified when the same text is written again. However, this kind of indentification would not offer any advantages over a verification system where all the users write different text (username and password). The user has to write some predefined text to be identified, so why not use different texts known only for individual users (passwords) and change harder identification task to a more straightforward verification task?

So, identification in this case is mainly useful for constant monitoring. Identification with keystroke dynamics is implemented by using a background task for collecting keystroke dynamics profile of the user's typing. Such system is not limited by short texts, but on the other hand there is no possibility of using only some predefined texts for identification. Thus, more general keyboad dynamics statistics have to be gathered. For example latencies between keys in all different key-pairs can be gathered – what is the average latency between A and B when the user writes AB and so on for all key-pairs.

"Continuous authentication by analysis of keyboard typing characteristics" by S.J. Shepherd is a study of a continuous authentication scheme. The goal of the study was to develop a system which could identify the user in as short time as possible, so that the potential damage an unauthorized user would be able to do would be minimized. In previous studies it had been noted that even 100 characters was enough for identifying users with a well designed system. The problem with identification with a very short text is that legitimate users can have temporary typing anomalies – speaking in a phone at the same time, for example – and the system should not be too sensitive. In the study it

was decided that forcibly logging out the user was too disruptive and the system only continuously evaluates the likelihood of the user being who he claims to be. If there is serious discrepancy then the administration can be warned and they can decide if further action is needed.

In the experiments the authors had promising results when measuring only latencies between keystrokes and durations of keystrokes and calculating the averages and variances globally. They had only four persons included in the tests, but at least between them even these very simple features were good enough for identification of users.

3.3 Commercial system: BioPassword

BioPassword[8] is a patented[9] user authentication system by an US company, BioNet Systems[10]. The company is better known for its Internet filtering product NetNanny which can be used to prevent Internet user's from accessing certain web-pages.

BioPassword is designed to replace the default log-in system used in Windows NT/2000/XP operating systems. It is not limited to a single computer, but it can be used in a Windows network environment where user accounts are stored centrally. Also a development kit exists which enables easy implementation of a similar authentication system in other applications. BioPassword works as follows:

- BioPassword software is installed on the server and the client workstations.
- The user must enroll to use their workstation and the network. Enrollment is done by typing the username and the password several times, 15 by default. The biometric template is stored on the server.
- After enrollment, the user can log in normally by typing the username and the password.
- The system checks the login attempt against the stored template. Only a user whose typing pattern matches the stored template is allowed to log in.

According to the patent[9] BioPassword is a quite straightforward implementation of keystroke dynamics systems implemented in many scientific studies [1, 4–7]. It uses both latencies between keystrokes and keystroke durations for verifying users. The patent does not reveal details on the used method for comparing measurements of a new login attempt to a stored profile. According to the patent the method stores the profiles of user's in a way that would be usable for identifying users continuously. However, continuous verification or identification is not apparently used in the product.

BioPassword is reviewed in [11] from a functional point of view. BioPassword was installed on a Windows 2000 server and a workstation. After few hassles during installation system was working properly and new user accounts could be added. When logging in for a first time, the new user was asked to write the username and password combination 15 times. After that, the log-in procedure worked just like default log-in dialog in Windows. Of course, the most important

part is whether keystroke dynamics add security. The reviewers tested whether they could log into each others accounts when they knew the username and the password. With the default security setting they were not able to log in to other person's account even after watching the other person writing the login information and trying to mimic the typing patterns. With a lower security setting logging to other person's account succeeded. On the other hand with the highest security setting even logging to one's own account was not always possible. For example, when writing the login information when standing up instead of sitting down logging in did not always succeed.

In another review[12] also few caveats were found. First, it is possible to bypass BioPassword by using *RunAs*-functionality after logging in normally. When using *RunAs* only the username and the password are needed, typing template is not checked. The *RunAs*-functionality can be disabled preventing this method of bypassing BioPassword, though availability of the *RunAs*-service is useful in some cases. Second, if 100% compliance is wanted and all the user accounts use BioPassword, then losing administrator access to the network is a possibility. If there is only one administrator account and the person using the account, for example, has an accident and breaks a finger, he won't be able to log in because the typing pattern without one finger will be different to the stored template. The problem is reduced if there are many administrators, which is the normal case.

On the whole, the reviewers were quite content with the BioPassword system. It was noticed to be non-obtrusive for the network and the users, it does not need any special hardware and is reasonably priced. The caveats, however, show that it is not easy to cover all potential routes for gaining access to user accounts. The problem would be considerably worse if the environment was more heterogenous, i.e, several different operating systems are used instead of only Windows. Using other operating systems than Windows is not currently possible is not currently possible with BioPassword.

4 Timing attacks on secure communications

When the user's keystroke dynamics profile is known, it may be possible to guess what is being written when only the latencies between keystrokes are measured. If the communication protocol sends every user-written letter separately, then it is possible to record their timings. Widely used SSH is an exaple of such protocol. From the timing-data it might be possible to guess which letters were written, or – in case of cracking passwords – at least reduce the number of possible password choices to crack with usual brute-force methods. Additionally, instead of using the keystroke dynamics profile of a specific user, the profile can be exchanged by a general approximate profile usable for all touch-typists.

Timing analysis of keystrokes and timing attacks against SSH[13] have been studied in [14]. SSH provides a secure encrypted communications channel between two hosts. However, there are two weaknesses despite used strong cryptographic algorithms: 1) The transmitted packets are padded to eight byte length,

which reveals approximate size of the original, unencrypted data, and 2) In interactive mode individual keystrokes are transmitted in separate IP-packets immediately after the key is pressed. Actually the initial login to a remote site using SSH does not leak timing information to the network, because the initial login sends the whole password in one packet. The timing information is leaked when an established SSH connection is used, for example, to change to super-user account and writing the super-user's password.

First, the IP-packets and their timings when writing password have to be recognized from the network. An example of packet sequence in the network when using "su" command to change to super-user account is presented in Figure 4. The problem of noticing important packet sequences is made easier by the fact that all the normal keystrokes sent to the SSH-server generate a returning packet because the character is echoed to the screen, but when writing a password characters are not echoed and consequently packets are send only to one direction, from the client to the server. This fact makes easier to notice when the user writes a password.

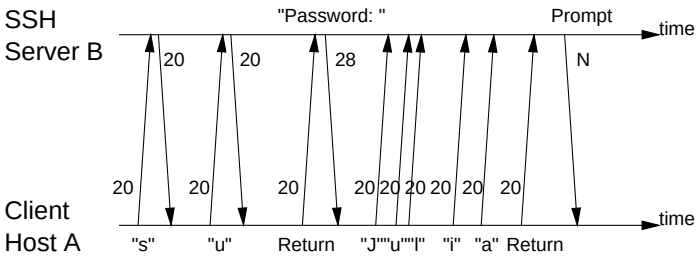


Fig. 4. Sequence of packets when using "su" command when changing to super-user. [14]

Before actually guessing the password based on intervals between characters there has to be information on what kind of latencies are to be expected between different keys. The authors studied intervals between many different key combinations by dividing key-pairs to several categories based on whether they are typed with alternating hands and are they both letter or number keys. 142 key pairs were chosen and latencies when typing them were measured. The results for different categories are in Figure 5. For example, all key pairs that are written using both hands had latencies <150 ms. So, if the attacker notices that there was larger than 150 ms latency between two packets, the keys were probably pressed using the same hand.

Distributions of intervals were noticed to follow Gaussian distribution, so they can be modeled with only average and standard deviation. From the latency information the authors created a Gaussian model for all key pairs. Gaussian distributions are presented in with the information gain as a function of latency

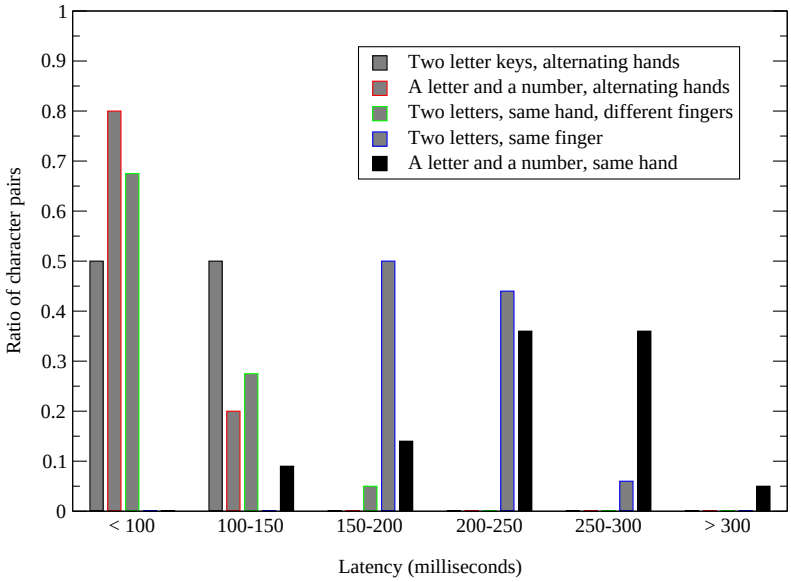
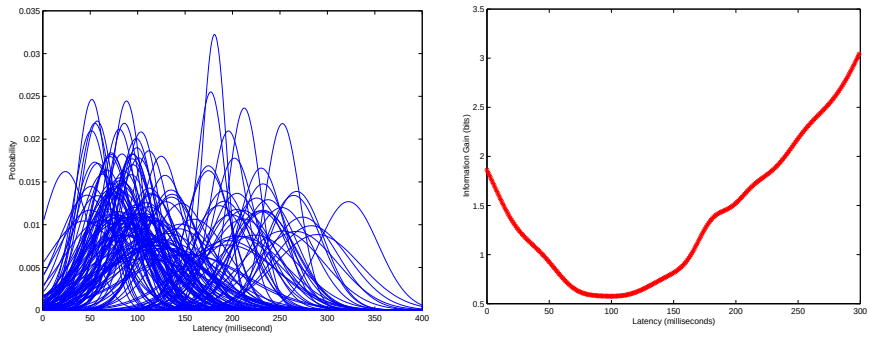


Fig. 5. Histogram of latency of key pairs. [14]

between two keystrokes in Figure 6. Information gain is an upper bound for how much information an attacker can extract from the timing information. With uniform distribution of characters the information gain is about 1.2 bits per character pair. Compared to entropy of written English, 0.6-1.3 bits per character [15], information gained from latency information is significant, even though for passwords entropy should be considerably higher than for normal English text.

The relation between latencies and character sequences is modeled as a Hidden Markov Model. A Markov Model is a way of describing a stochastic process where a transition from a state to the next state is dependent only on the current state. In Hidden Markov Model the current state cannot be directly observed, only some outputs of the state are observed. Observed outputs can then be used to infer the prior path of the process. In this case the character pairs are used as (hidden) states, and the latencies as observed outputs. The Viterbi algorithm can be used to solve the most likely sequence of states for certain sequence of observed latencies from the Hidden Markov Model. In this case the latency distributions of difference characters overlap highly, so the probability that the most likely sequence is the correct one is very low. Because of that, the authors enhanced the algorithm to provide n most likely sequences. The enhanced algorithm is called n-Viterbi.

In the experiments 8-character long passwords with random characters were tried to be guessed by collecting real timing data of a user writing the password.



(a) (b)

Fig. 6. (a) Estimated Gaussian distributions for all measured 142 key pairs; (b) Information gain as a function of latency between two keystrokes. [14]

As discussed earlier, it is very unlikely that the first sequence suggested by n-Viterbi algorithm is the correct one. So, the success of the method is measured by how large part of the total password space – all possible combinations of 8 letter and number characters – had to be tried before finding the correct one. The results are in Figure 7. The tests indicate that on average the correct password is found after trying 2.7% of the password space and the median is 1.0%. This means that compared to a brute force search, which would need to search 50% of the password space on the average, there is almost a 50-fold decrease. This is quite significant since testing the whole password space could take months, but using the described timing attack the time needed would be reduced to days.

The same attack type is applicable for all similar protocols. The authors propose few countermeasures. First, the problem that writing a password is quite easy to notice from the network because there are no returning packets could be fixed by sending dummy packets back. Timing data could still be gathered but the password would not stand out so obviously from the packet stream. Alternatively timing data could be messed up by adding random delays when sending packets, but the delays would have to be in order of several hundred milli-seconds to be effective. Such large delays would be quite irritating for the user. Additionally random noise will be canceled out if the attacker can listen the network for a long time. Another potential fix would be to send dummy packets continuously, so the attacker would not know which packets contain real user-written characters. This would waste some bandwidth.

SSH is not the only vulnerable protocol. Almost every type of secure remote connection protocols sends keystrokes just as the user writes them and the same security implications apply to all of them. Then again, there are lot of other data potentially going through, for example, a VPN (Virtual Private Network) connection besides characters of the pressed keys so recognizing which packets are relevant keystrokes could be challenging. Recognition of relevant packets is

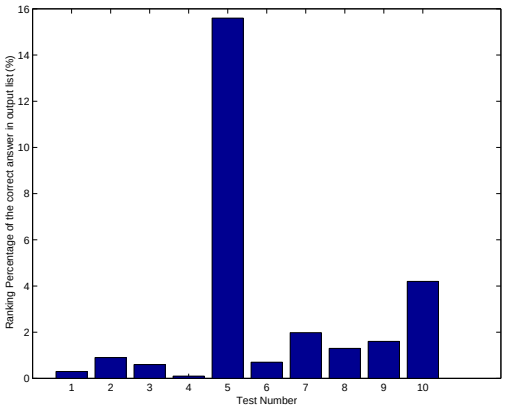


Fig. 7. The percentage of the password space tried by Herbivore in 10 tests before finding the password. [14]

mandatory for measuring timings between them, so if the relevant packets are hard enough to notice them, then the attack fails.

5 Conclusions and discussion

Keystroke dynamics is a very cheap biometric verification method because there is no need for any additional hardware besides a normal keyboard. On the other hand, keystroke dynamics is not a strong biometric identification method, so it is not applicable to situations where most stringent security measures must be followed. Using keystroke dynamics makes a username/password-based authentication procedure significantly more secure. Knowing the username and the password is not enough for logging in to the system, also the typing dynamics of the real user has to be imitated. The caveats of BioPassword on the other hand demonstrated that it is not easy to apply such restrictions to all possible routes for gaining access on user accounts. Keystroke dynamics could also be used in, for example, cash dispensers when the user writes the PIN-code.

There are some problems in keystroke dynamics based authentication systems which seem to be often by-passed with brief mention [6], if any, on articles. Many of the application papers use either normal words or slightly longer phrases than what is customary with passwords. If the password is used only in settings where the keystroke dynamics are also checked, for example, there is no chance to bypass the keystroke dynamics phase when logging in from a networked computer elsewhere, then it might be good enough to use a simple word as a password. Existing words are otherwise too easy to crack by dictionary attacks. Longer phrases might not be too easy for dictionary or brute force attacks, but they are problematic with many old (and some new) systems which support only classic 8-

character passwords. In case of using keystroke dynamics with normal passwords (up to 8-characters long with small and capital letters, numbers and maybe even punctuation), the user's writing is often quite erratic at least shortly after changing to a new password to be very usable in this kind of system. Typing such password tends to be quite slow before learning to write it "instinctively". Thus, when the system learns user's typing pattern when the password is changed, the user writes the password quite slowly and erratically because there are several hard-to-type characters. After a few weeks there is no more need to stop and think what character is next so the speed of typing will be much faster. On the other hand, the commercial system, BioPassword, seems to work reasonably well according to independent if not very comprehensive tests.

Most of the computers are used in configurations where it is possible for the user to change peripheral devices, including the keyboard. A serious attacker would be able to create a device which looks like a normal keyboard to a computer but sends pre-defined key-sequences to the computer. So, if a cracker knows a username and the password, he might be able to create beforehand a key-sequence including correct timoigs by studying habits of the real user. Using the device and the timings the attacker could gain access to the system. Even this kind of attack would be made significantly harder if a continuous keystroke dynamics identification was used. However, there does not appear to be such systems available.

As for attacks on secure communications there are no solutions without drawbacks: either there must be additional random delays when sending packets which is annoying to the user or bogus packets are sent which wastes some bandwidth. SSH was noticed to be easy target for certain cases, one of the reasons being that it is quite easy to notice from the encrypted data when the user is writing a password – no packets come back from the server because password characters are not echoed to the screen. On the other hand, this kind of attack requires the attacker to know keystroke dynamics of the user whose connection is being eavesdropped, unless the user uses standard touch-typing system. With standard touch-typing system the timings can be estimated well enough for purposes of snooping passwords from other touch-typists.

References

1. Obaidat, M.S., Sadoun, B.: Verification of computer users using keystroke dynamics. IEEE Transactions on Systems, Man and Cybernetics **27** (1997) 261–269
2. Miller, B.: Vital signs of identity. IEEE Spectrum **31** (1994) 22–30
3. Shepherd, S.J.: Continuous authentication by analysis of keyboard typing characteristics. In: European Convention in Security and Detection, Brighton, UK, Bradford University (1995) 111–114
4. Bleha, S., Slivinsky, C., Hussien, B.: Computer-access security systems using keystroke dynamics. IEEE Transactions on Pattern Analysis and Machine Intelligence **12** (1990) 1217–1222
5. Lin, D.T.: Computer-access authentication with neural network based keystroke identity verification. In: International Conference on Neural Networks, Houston, Texas, USA (1997) 174–178

6. Coltell, O., Badfa, J.M., Torres, G.: Biometric identification system based on keyboard filtering. In: IEEE International Carnahan Conference on Security Technology, Madrid, Spain (1999) 203–209
7. Haider, S., Abbas, A., Zaidi, A.K.: A multi-technique approach for user identification through keystroke dynamics. In: IEEE Internation Conference on Systems, Man, and Cybernetics. Volume 2., Nashville, TN. USA (2000) 1336–1341
8. BioPassword. [Website] (retrieved October 23, 2003) From: http://www.biopassword.com/.
9. Zilberman, A.G.: Security method and apparatus employing authentication by keystroke dynamics (1998) United States Patent 6,442,692.
10. BioNet Systems, LLC. [Website] (retrieved October 23, 2003) From: http://www.bionetsystems.com/.
11. Altman, A.: Review of BioPassword 4.5. [HTML-document] (retrieved October 23, 2003) From: http://www.biometritech.com/features/022502review.htm.
12. Bragg, R.: Biometric security products. [HTML-document] (retrieved October 24, 2003) From: http://www.mcpmag.com/Features/article.asp?EditorialsID=270.
13. Ylonen, T., Kivinen, T., Saarinen, M., Rinne, T., Lehtinen, S.: SSH Protocol Architechture. IETF Internet-Draft. (2003) From: http://www.ietf.org/internet-drafts/draft-ietf-secsh-architecture-14.txt.
14. Song, D.X., Wagner, D., Tian, X.: Timing analysis of keystrokes and timing attacks on SSH. In: 10th USENIX Security Symposium, Washington, D.C., USA (2001) 337–352
15. Shannon, C.: Prediction and entropy of printed english. Bell Systems Technical Journal **30** (1951) 50–64