User Authentication and Identification Based on SSH Packets

Ole André Hauge[1]

Abstract: Secure Shell (SSH) is one of the most used protocols providing encrypted remote access to web servers, cloud-based Linux-derived machines, and a plethora of network equipment. It is established as a secure solution, and any vulnerabilities in the protocol experienced today tend to be related to implementation issues. However, cryptanalysis attacks remain prominent, and as the research field of biometrics evolves we discover new ways of exploiting the design of encryption protocols. When SSH is used in interactive mode, each key-press is immediately transmitted as an encrypted IP packet. This means that capturing the packets for a secure shell session reveals the biometric feature that is the latency between each keystroke. In this paper we confirmed, using a limited dataset and five different anomaly-detection algorithms that we should be able to use this timing feature to authenticate and identify users. We further noticed that environmental factors might not affect the quality of the inter key-stroke latency features extracted from keystroke samples recorded to a concerning degree in contrast to previous beliefs.

Keywords: Secure shell, SSH, interactive mode, timing attack, Euclidean distance, Manhattan distance, k-nearest neighbors, support vector classification, keystroke dynamics

1 Introduction

Secure Shell (SSH) is one of the most used protocols providing encrypted remote access to web servers, cloud-based Linux-derived machines, and a plethora of network equipment. As SSH has existed since 1995 [YCL06], the open-source protocol has had time to be properly tested and improved. Any vulnerability in the protocol experienced today tends to be related to implementation issues. However, cryptanalysis attacks remain prominent and as the research field of biometrics evolve we discover new ways of exploiting the design of encryption protocols.

Keystroke dynamics (KD) refer to the timing features that are collected from a person typing at a keyboard. In biometrics, KD is viewed by most as a behavioral characteristic although some dispute the uniqueness of the features [TGG13]. The keystroke rhythm of users is recorded and used to create biometric references that can be used to recognize the user when presented with a probe in the future.

Keystroke dynamics can extract many features from persons typing behavior. The most common features are timing information; key downtime and key uptime, force, sound, and choice of fingers, most of which require access to the computer and/or a special type

[1] Faculty of Information Technology and Electrical Engineering at the Norwegian University of Science and Technology (NTNU), Department of Information Security and Communication Technology, Teknologivegen 22, 2815 Gjøvik, Norge, oleahau@stud.ntnu.no

of keyboard to capture the samples. This is usually done with keystroke loggers. These features enable a deeper understanding of the person that is typing. Research has proven that a person's emotional state [MK20],[Ko18], age, and gender can be derived from their typing characteristic [Pe17],[TAK18]. That being said, a person's typing is prone to change due to external factors like the type of keyboard, room lighting, location of the keyboard, emotional state, and timing issues in online data capture [BW12],[Za17].

When SSH is used in interactive mode, each key-press is immediately transmitted as an IP packet. This means that capturing the packets reveals the biometric feature that is the latency between each keystroke. This is said to be one of the more prominent of the timing features [SWT01] of the most important and most researched features regarding keystroke dynamics [Bo21, p. 26]. This paper will look at whether this specific timing feature can be used to authenticate and identify users of SSH in interactive mode.

The rest of the paper is structured as follows: In section 2 the anomaly-detection algorithms and related work are explained. Section 3 presents how the data acquisition process, data processing, and comparison were performed, while the results of the data analysis and processing are presented in section 4. The methods and results are further discussed in section 5 before the paper is concluded and future work is proposed in section 6.

2 Background and Related Work

Anomaly-detectors are, in the context of biometrics, used for comparison of probes and references to measure similarity or dissimilarity. Detectors differ in the choice of parameters, once these parameters are selected a detector can compute a comparison score for a presented probe, which can be used to authenticate or identify a person. This paper uses the Manhattan-, scaled Manhattan-, and squared Euclidean distance for authentication, while KNN and SVC are used for identification. Information about the anomaly-detectors used in this paper is listed bellow.

2.1 Euclidean Distance

Euclidean distance is used to calculate the shortest straight-line distance between two points. These points can be represented as vectors, thus resulting in a comparison score that reflects the difference between the two. Squaring each value in the equation makes the comparison score more sensitive to changes in the input values, which can be a desirable attribute when looking for similarity or dissimilarity in vectors (1) [Ba05, p. 394],[Sh15].

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{1}$$

2.2 Manhattan Distance

Manhattan distance is used to calculate the shortest distance, in a grid-like manner, between two points (2). As with the Euclidean distance, these points can be represented like vectors and result in a comparison score [Ba05, p. 395],[Sh15]. Scaling the Manhattan distance is done by dividing the absolute values by the standard deviation, which makes it more sensitive to changes in the input (3) [Ar05].

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i| \tag{2}$$

$$d(x,y) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{a_i} \tag{3}$$

2.3 K-Nearest Neighbor Classification (KNN)

KNN is a simple classifier that generally delivers good results for smaller datasets. The model is trained to classify inputs based on a plurality vote of its neighbors. Presenting the model with a probe returns a comparison decision as a match or non-match [RSN10].

2.4 Support Vector Classification (SVC)

The SVC machine learning model (ML) is the Python implementation of the SVM ML model. SVC is trained on mated data. This data is mapped into a vector space, and a hyperplane is fitted to the data to work as a decision boundary, depending on the dimension of the data. Probes presented to the ML model are mapped into the same vector space and comparison decisions are given based on the data's relative position to the decision boundary [No06].

2.5 Related Work

Significant research related to this subtopic of keystroke dynamics are related to timing attack on SSH [SWT01], accuracy of anomaly-detectors algorithms [KM09], mobile keystroke dynamics [MKC20], authentication [Ga80],[BW12], identification [BW12], and the use of neural networks and machine learning [MGP17],[HGS08]. This paper focuses on how the inter-keystroke latency can be analyzed to authenticate and identify users, and specifically uses flaws in the SSH transmission leaking latency information that can be analyzed.

Another study focuses on time-frequency analysis for user authentication with good results [TA21]. Ramin Toosi et. al. used dynamic time wrapping and winger distribution

to calculate the similarity measure between an input sample and user reference samples by obtaining the time-frequency representation of the signals before comparing the two signals in the time-frequency domain.

Saket Maheshwary et. al. developed a neural network (NN) architecture to authenticate or identify different users based on their keystroke characteristics. Their approach was up to 10 times faster compared to previous neural network models. With an Equal Error Rate (EER) of 0.030 for the user authentication task and overall accuracy of 93.59% for user identification, they proved that the NN is a viable solution [MGP17]. This is similar to the work done in this paper, but in contrast, they used more than one feature to authenticate and identify users.

A closely related study [SWT01] has proven that the SSH protocol leaks information in two ways when timing attacks have been applied. In some configurations, SSH only pads packets by an eight-byte boundary, thus leaking the approximate size of the original data. Furthermore, every keystroke is transmitted immediately in separate packets when in interactive mode, enabling an eavesdropper to learn the inter-keystroke latency of the user. Dwan X. et. al. show that the information leaked resulting from these 2 vulnerabilities is approximately 1.2 bits of information gain per character pair, which is significant in the context that the entropy of written English is between 0.6-1.3 bits per character [Sh51]. Although their study is closely related to this study, they look at how the padding and inter-keystroke latency leak information about what is written, while this paper focuses on authenticating and identifying users using only the inter-keystroke latency.

3 Method

This paper aimed to capture and analyze the network traffic of subjects in interactive mode to see if it was possible to authenticate and identify users. An acquisition process was used to produce good samples for the feature extraction process. The inter-keystroke latency features were used to create a reference for the system. The comparison was done using five different anomaly-detection algorithms before using various performance metrics to validate the hypothesis. This was done in three main stages: Capture process, preprocessing, and processing, which is explained in further detail in the sections below.

3.1 Capture Process

The SSH traffic was captured from subjects of various gender and technical background to give a realistic representation of the data. The test environment consisted of a client-to-server setup. The subjects were given a set of instructions (see appendix B) to follow. They were asked to open an SSH connection to the server. The setup had Wireshark as a capturing device running on the server capturing all the network traffic related to the SSH connections as seen in figure 1.

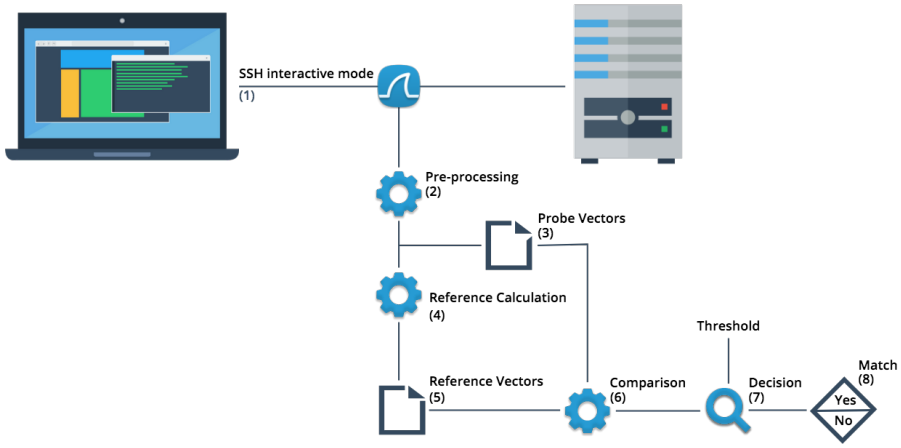

Fig. 1: The Experiment Setup

The dataset for the anomaly-detector algorithms consisted of 12 files, where each file contained the captured data from a single capture subject. Each subject typed the word "welcome42" + "enter" 30 times.

The dataset for the machine learning algorithms was a database of 1350 (5 x 27 x 10) inter-keystroke latency features, created consisting of data from 5 capture subjects. It had the subjects' features listed rowwise resulting in an 11x135 matrix, including the subject ID in the first column.

3.2 Pre-processing

Erroneous data containing typos and typing corrections which added extra timing latency was removed. It was located with the terminal history files that were stored from when each user typed in the SSH session, combined with the difference in length of the packet sequences. Sequences that were longer than 10 packets were discarded as errors without any more comprehensive analysis. Figure 2 illustrates how a packet sequence looked. Furthermore, excess information produced in the packet transmission, like the Diffie-Hellman key exchange and exit-sequence information, was removed.

When the data was cleaned it was converted from pcap files to text files with a *"Action|time|value"* format. The start and stop of samples were indicated by *"-|sampleNr|-"*. As most subjects had a unique IP address it was possible to use this attribute to separate the data. A couple of subjects used the same IP address, but since SSH connections are initialized with a TCP connection, the unique port that was assigned to each subject was used to differentiate the sessions.

The removal of errors meant that the number of samples per subject was uneven. To adjust this, the number of samples per user was brought down to 27 by removing the last samples

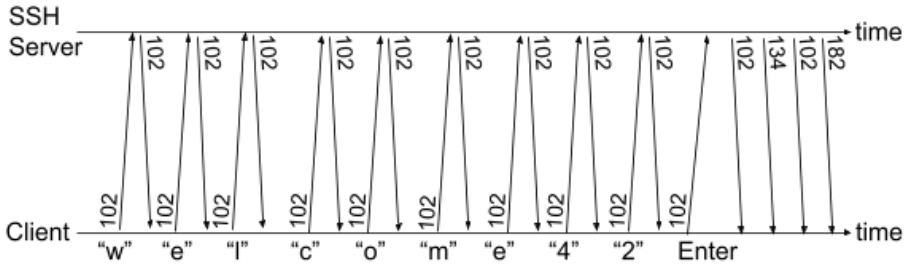

Fig. 2: The traffic signature associated with writing "welcome42" and enter in an SSH session. The numbers in the figure are the size (in bytes) of the corresponding packet length.

in the files until all files had the same length. Three subjects had to be removed from the analysis as there were too many errors in the collected data.

The rest of the processes for the three anomaly-detector algorithms were done using MATLAB, while Python was used to implement the ML algorithms. The data was loaded into MATLAB reading each of the files in the dataset and storing the information in variables. In Python, the data were loaded using the database.

3.3 Processing and Analysis

The processing and analysis of the data were done in three steps: feature extraction, reference creation, and performance analysis. These steps are described in further detail below.

3.3.1 Feature Extraction

The inter-keystroke latency features were extracted from the intermediate samples that were stored per capture subject in a feature variable. The ML models were fitted to the training data, which would be used as a reference when comparing to probes. The next step was only conducted for the anomaly-detectors.

3.3.2 Reference creation

The reference for the anomaly-detectors was created with 20 of the 27 samples for each user by calculating the mean (mu) and standard deviation (sigma) over the inter-keystroke latencies for each user. The reference used 20 samples as testing proved this to produce the best EER result for the system.

3.3.3 Performance analysis

The performance analysis was conducted in four steps and is described in more detail below followed by a short explanation of how it was implemented in the MATLAB code and how it was performed by the machine learning algorithms.

1. **Select anomaly-detectors:** The choice of the anomaly-detectors was based on the research of [KM09]. Five anomaly-detection algorithms that showed promising results were selected (see table 1). The squared Euclidean-, Manhattan-, and scaled Manhattan-distance was used for testing the authentication, while KNN and SVC were chosen for testing the identification. Their results are further strengthened by other researchers [BW12].

2. **Calculate the genuine and impostor scores:** The anomaly-detectors for the authentication system were implemented according to the aforementioned formulas for mated and non-mated comparison trails. This was done with one-to-one comparisons of the 7 probes per user and reference, resulting in comparison, genuine, and impostor scores.

3. **Calculate FMR and FNMR:** The False Match Rate (FMR) and False Non-Match Rate (FNMR) were used to determine the performance of the biometric authentication system. FMR is the proportion of impostors inaccurately matched to genuine users, while FNMR is the proportion of genuine users rejected from the system [bi14].

4. **Calculate EER:** The Equal Error Rate (EER) of the system was derived using the FMR and FNMR results. EER is where FMR equals FNMR and represents a good operating point of the system. This can be further tweaked to the system operator's needs. Researchers in the field often use EER as the performance score. The lower the score, the better the system [bi14].

Nr.	Detector	EER	Nr.	Detector	zero-miss false-alarm rate
1	Manhattan (scaled)	0.096	1	Nearest Neighbor	0.468
2	Nearest Neighbor	0.100	...	...	...
...	...	...	4	SVM (one-class)	0.504
4	SVM (one-class)	0.102	5	Manhattan (scaled)	0.601
...	...	...	...	...	...
10	Euclidean	0.171	10	Euclidean	0.875

Tab. 1: Quality of the inter-keystroke latency dataset [KM09]

The MATLAB script CalculateScores.m was used to calculate the comparison for the genuine and impostor scores. This is where the different anomaly-detectors for authentication were implemented and calculated. Outliers of one second and more were removed when calculating the distances to improve the results. The EER was calculated with the GetEER.m where the intersection of the genuine and impostor scores was calculated.

The machine learning algorithms used the previously described reference database, setting aside 80% of the data for training and 20% for probes. They returned recall, precision, and f1-scores that were used to validate if identification was possible [RSN10].

Andreas C. Muller et. al. define the precision metric as a "[measure of] how many of the samples predicted as positive are actually positive [...]", while recall is as a "[measure of] how many of the positive samples are captured by the positive predictions [...]. The recall metric is used as a performance metric when we need to identify all positive samples; that is, when it is important to avoid false negatives." The f-measure is used to get a greater impression by summarizing the two aforementioned metrics as the "[...] harmonic mean of precision and recall. [MG18, p. 289]". The greater the f1-score, the better is the performance of the ML-model [RSN10].

The implementation and performance testing was done using the Python and MATLAB programming language, of which the files are available in appendix A.

4 Results

The result of the data capturing was a reference database consisting of 3240 inter-keystroke latency features, after removing erroneous data. The samples per user were restricted to 27 to get the same amount of data, as described in section 3.2. The anomaly-detector algorithms proved to be efficient, considering that only one timing feature was used.

4.1 Data Quality

The result of the acquisition process is shown in Table 2, represented by the total average, variance, and erroneous data of the capture subjects typing of "welcome42" + "enter". Most subjects proved to have a consistent typing pattern as seen in the low variance values. However, it is evident by the high erroneous scores that the acquisition process has room for improvement.

The EER scores for squared Euclidean, Manhattan, and scaled Manhattan distance were calculated using the datasets of the 12 users and are presented in table 3. The precision-, recall-, and f1-scores for the machine learning (ML) models were calculated with a smaller dataset consisting of the data for users 1-5. For a more comprehensive comparison between the anomaly-detection algorithms and the ML models to be conducted, the models should be fitted to a larger dataset. However, these results still indicate better results using KNN or SVC.

It is worth noting that the EER score is used to describe the performance of the entire biometric system used for authentication, while the f1-score only represents the performance of the machine learning models used for identification.

Looking at the results it is evident that squared Euclidean distance was the best of the anomaly-detection algorithms for authentication. Furthermore, it is observed that the KNN and SVC models perform well for identification, where SVC shows a slightly better result.

User	Average	Variance	Erroneous Data (%)
1	0.0826	0.0087	0
2	0.264	0.356	0
3	0.387	0.207	0
4	0.271	0.233	9.9
5	0.335	0.818	0
6	0.223	0.0181	0
7	0.207	0.0373	0
8	0.26	0.098	0
9	0.117	0.0323	0
10	0.0875	0.0105	0
11	0.074	0.00947	36
12	0.109	0.00569	0
13	NA	NA	100
14	NA	NA	100
15	NA	NA	100

Tab. 2: Quality of the Inter-keystroke latency references

Algorithm	EER	Training Samples	Probes
Manhattan distance	0.2930	20 (per user)	7 (per user)
Scaled Manhattan distance	0.2313	20 (per user)	7 (per user)
Squared Euclidean distance	0.2262	20 (per user)	7 (per user)
Algorithm	**F1**	**Training Samples (%)**	**Probes (%)**
KNN	0.8467	80	20
SVC	0.9216	80	20

Tab. 3: Accuracy Scores

4.2 Machine Learning Models

After training the KNN model with the reference database, the best KNN f1-score was 0.8467, while the SVC model got an f1-score of 0.9216. Both used an 80/20 training-test split of the data. Figure 3 and 4 show the corresponding similarity matrices of the comparison. The similarity matrices show the comparison results where each reference is compared to the other users' data. In figure 4 we can, for example, see that none of the other users' probes match with the reference of user one, which means that there are no false matches for user one.

Looking at the two similarity matrices it is evident that the SVC model, with the given prerequisites, is better at identifying a user. This is made further evident by the difference in shading of the matrices, as figure 4 show fewer colored areas compared to figure 3.

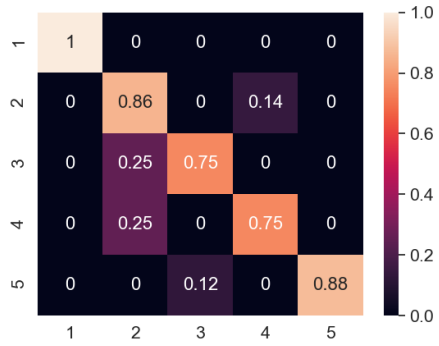

Fig. 3: Similarity Matrix for KNN

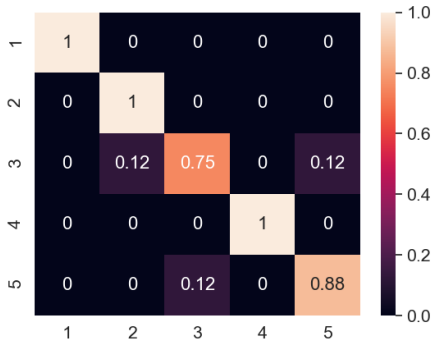

Fig. 4: Similarity Matrix for SVC

5 Discussion

5.1 Acquisition Process

The time given for the research and work-related travels affected the method of the acquisition process, which is one of the most important aspects of the results. Thus, the captured data had more errors than intended, which had to be removed. Wrongfully typing the word would negatively affect the comparison scores as the reference- and probe-vectors might end up being of different sizes and contain disproportionate feature values.

A way to avoid this would be to improve the acquisition process by writing a program asking the capture subjects to type the word and only accept their input if it was typed correctly. That would include limiting their ability to try to correct their error with backspace, as implemented by Thomas Fluke in his master thesis [Fl20]. Implementing this would probably result in better raw data, which would improve the preprocessing and processing of the data later.

Instead, the capture subjects were left to conduct the capture process following a set of instructions. Some subjects consecutively typed the word wrong (welcom42) for the entire duration of the capture process and had to repeat the task. Others used the arrow keys to quickly reuse the last typed command. If the acquisition process were to be conducted again these errors should be eliminated using the above-mentioned solution, although some of these errors were confirmed to be due to the subjects' quick reading of the instructions.

The second limitation of the acquisition process was the number of subjects and the number of data captured. Different methods were considered for the data capture, one of which was to install an Open-SSH Server on AWS and create a wrapper script for an SSH installation that the subjects could download and use. The idea was that the wrapper implemented the functionality mentioned above. However, due to time and monetary resources, this so-

lution was not used. The results would have greater accuracy and credibility if there had been time and the possibility of including more subjects in the research.

5.1.1 Environment

Different environments were used when capturing data, due to work-related travels abroad during large parts of the time spent writing the paper. As mentioned in section 2, this means that the results might have been affected negatively due to environmental changes. However, the results of the comparison scores are similar to the results of others [BW12],[KM09], and we see that the KNN and SVC models are surprisingly positive at 0.8467 and 0.9216 respectively, implying that the changes in the environment might not have that great of an effect on the results as previously believed [Za17],[BW12].

5.1.2 Choice of Word

The length of the word was chosen to give a sufficient amount of features without being too long, as the length could have resulted in a higher number of false typings of the word. The word had only lowercase to avoid meta keys like *shift* and *ctrl* which are not sent as IP-packets when pressed. Numbers were added as normal commands in the terminal usually are combinations of letters and numbers, thus the results might be closer to real-world scenarios. Actual terminal commands were not used due to the varying technical level of the capture subjects as their typing pattern might not have been accurate for combinations like “ls -la” or “ps -ef | grep tty”. welcome42 is hopefully a more neutral way of capturing the samples for the SSH sessions.

The subjects’ familiarity with the word and the fact that none of the subjects used English as their primary language could have affected their typing pattern. Due to the time limitations, the subjects were not given time to learn or get familiar with typing “welcome42”, which might have improved their consistency even more.

5.2 Pre-processing

The removal of excess and erroneous samples during the pre-processing might have resulted in the unintended removal of inter-keystroke data, which could have affected the comparison scores. Since no good guides or patterns exist for the pre-processing of SSH packet traffic in this context, the patterns had to be created first.

5.3 Processing

All anomaly-detectors have strengths and weaknesses. Euclidean and Manhattan distance has two weaknesses: (1) data vectors with no common data values may result in smaller

distances compared to the other pairs of data, (2) the largest scaled data value dominates the others. The first is solved by cleaning the dataset ensuring that only relevant features are used in the comparison process. Normalizing the data also prevents large values from dominating the result [Sh15].

A weakness of using the KNN model, especially when handling heterogeneous data, is that the performance often is affected by ad-hoc choices of similarity metrics [YR06]. Since the reference databases in this paper are homogeneous the results should not be affected by this.

As for the SVC model, the biggest vulnerability is over-fitting of the data, where the ML-model is trained to accurately categorize the training data, but with no guaranty of being good at categorizing new data that is not included in the training data [CT10]. To avoid this the SVC model was trained with an optimized C-value, the penalty parameter for the error term, as increasing C-values may lead to over-fitting [No06]. The ML model compared new probes that were not part of the reference databases resulting in an unbiased classification. A biased classification e.g. no-detection of over-fitting would have happened if the ML model compared samples taken from the reference databases.

5.3.1 Best Anomaly-Detection Algorithm

The results show that the SVC model was the most accurate anomaly-detector for the identification of the ones tested in this paper. Kevin et. al. concluded that K-Nearest Neighbor should perform well as an anomaly-detection algorithm [KM09]. Their implementation of an SVM model got fourth place in both score categories. The difference between their findings and the ones of this paper is most likely due to the number of samples captured, the type of features that were used, the preprocessing of the samples, and the implementation of the detectors. More captured data could reveal discrepancies and achieve more accurate comparison and accuracy scores.

The results also show that squared Euclidean distance was the best anomaly-detection algorithm for authentication, with a slightly lower EER than scaled Manhattan distance. This differs slightly from the result from other researchers [KM09] where the scaled Manhattan distance performed best. However, both Manhattan distance and scaled Manhattan distance prove quite accurate compared to results of previous researchers [BW12],[BW12]. If more time was available it would be interesting to look at different calculations of the reference used to calculate the comparison scores, and what effect they have on the results. This could be done by a one-to-many comparison splitting each subject's sample data into 22 references and 5 probes and then have a comparison decision finding a match or not. Another method could be to divide the samples into larger groups, calculate an average reference from that per group and do a one-to-many comparison.

6 Conclusion and Future Work

The results confirmed that it is possible to authenticate and identify a user by analyzing the packet samples that are recorded during interactive SSH connections, despite the use of only one timing feature and a database with 3240 timing features. Five different anomaly-detection algorithms were used to compare the data, three for authentication and two for identification, where the squared Euclidean distance and SVC model proved most accurate. This result further supports and strengthens the relevance of the inter-keystroke latency feature. Results further showed that environmental factors might not affect the quality of samples and probes that are captured to extract inter-keystroke latency features to a concerning degree in contrast to the general beliefs of [BW12],[Za17].

Further work should be done to investigate if authentication and identification of users of SSH in interactive mode can be done with higher accuracy. This could be done with larger datasets and better processing steps. One could also look into what can be done to eliminate these kinds of timing attacks against SSH connections, whether it is a revision of the protocol itself, additional features, or plugins. Other research could be conducted on the effect different calculation methods for the references have on the result of anomaly-detection algorithms.

7 Acknowledgements

I would like to thank my colleagues for setting aside time to partake in my data collection process. Without their help, the collection process would have been hard and tedious. Further thanks go to Sindre Asplem for aiding with knowledge and implementation of the machine learning algorithms

References

[Ar05] Araújo, Lívia; Jr, L.H.R.; Lizarraga, Miguel; Ling, Lee; Yabu-uti, João: User authentication through typing biometrics features. Signal Processing, IEEE Transactions on, 53:851 – 855, 03 2005.

[Ba05] Bailey, Donald G.: An Efficient Euclidean Distance Transform. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[bi14] Understanding Biometric Performance Evaluation, https://precisebiometrics.com/wp-content/uploads/2014/11/White-Paper-Understanding-Biometric-Performance-Evaluation-QR.pdf, Stand: 14.03.2021.

[Bo21] Bours, Patrick: Keystroke Dynamics [ppt.]. NTNU, 2021. Available on the IMT4126 Blackboard room.

[BW12] Banerjee, Salil P.; Woodard, Damon L.: Biometric Authentication and Identification using Keystroke Dynamics: A Survey. Journal of Pattern Recognition Research, 7:116–139, 07 2012. doi:10.13176/11.427.

[CT10] Cawley, Gavin C.; Talbot, Nicola L. C.: On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. Journal of Machine Learning Research, 2010. https://jmlr.csail.mit.edu/papers/volume11/cawley10a/cawley10a.pdf, Stand: 14.03.2021.

[Fl20] Flucke, Thomas: Identification of Users via SSH Timing Attack. The Faculty of California Polytechnic State University San Luis Obispo, 2020. https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=3587&context=theses, Stand: 14.03.2021.

[Ga80] Gaines, R. Stockton; Lisowski, William; Press, S. James; Shapiro, Norman: Authentication by Keystroke Timing: Some Preliminary Results. RAND Corporation, 1980. Santa Monica, CA.

[HGS08] Hu, J.; Gingrich, D.; Sentosa, A.: A k-Nearest Neighbor Approach for User Authentication through Biometric Keystroke Dynamics. In: 2008 IEEE International Conference on Communications. pp. 1556–1560, 2008.

[KM09] Killourhy, Kevin S.; Maxion, Roy A.: 2009 IEEE/IFIP International Conference on Dependable Systems Networks Comparing Anomaly-Detection Algorithms for Keystroke Dynamics. pp. 125–134, 2009.

[Ko18] Koakowska, A.: Usefulness of Keystroke Dynamics Features in User Authentication and Emotion Recognition. Human-Computer Systems Interaction: Backgrounds and Applications 4, pp. 42–52, 2018. Cham, ISBN: 978-3-319-62120-3, DOI: 10.1007/978-3-319-62120-3_4.

[MG18] Muller, Andreas C.; Guido, Sarah: Introduction to Machine Learning with Python, First release 2016, 4th ed. 12 2018. ISBN 9781449369415.

[MGP17] Maheshwary, Saket; Ganguly, Soumyajit; Pudi, Vikram: Deep secure: A fast and simple neural network based approach for user authentication and identification via keystroke dynamics. Proceedings of First International Workshop on AI in Security, 2017.

[MK20] Maalej, A.; Kallel, I.: Does Keystroke Dynamics tell us about Emotions? A Systematic Literature Review and Dataset Construction. In: 2020 16th International Conference on Intelligent Environments (IE). pp. 60–67, 2020.

[MKC20] M., Emanuele; K., Himanka; C., Patrizio: Mobile keystroke dynamics for biometric recognition: An overview. The Institute of Engineering and Technology, Allegheny College, 2020. https://doi.org/10.1049/bme2.12003.

[No06] Noble, William S: What is a support vector machine? Nature Biotechnology, 2006. https://doi.org/10.1038/nbt1206-1565, Stand: 18.05.2021.

[Pe17] Pentel, Avar: Predicting Age and Gender by Keystroke Dynamics and Mouse Patterns. Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization, pp. 381–385, 2017. New York, NY, USA, ISBN: 9781450350679, DOI: 10.1145/3099023.3099105.

[RSN10] Russell; Stuart; Norvig, Peter: Artificial Intelligence: A Modern Approach. Prentice Hall, 3 edition, 2010.

[Sh51] Shannon, C. E.: Prediction and entropy of printed English. The Bell System Technical Journal, 30(1):50–64, 1951. DOI: 10.1002/j.1538-7305.1951.tb01366.x.

[Sh15] Shirkhorshidi; Seyed, Ali; Aghabozorgi; Saeed; Ying, Wah Teh: A Comparison Study on Similarity and Dissimilarity Measures in Clustering Continuous Data. PLOS ONE, 10(12):1–20, 12 2015.

[SWT01] Song, Dwan Xiaodong; Wagner, David; Tian, Xuqing: Timing Analysis of Keystrokes and Timing Attacks on SSH. 10th USENIX Security Symposium (USENIX Security 01), 08 2001. Washington, D.C., https://www.usenix.org/conference/10th-usenix-security-symposium/timing-analysis-keystrokes-and-timing-attacks-ssh.

[TA21] Toosi, Ramin; Akhaee, Mohammad Ali: Time–frequency analysis of keystroke dynamics for user authentication. Future Generation Computer Systems, 115:438–447, 2021. DOI: https://doi.org/10.1016/j.future.2020.09.027.

[TAK18] Tsimperidis, Ioannis; Arampatzis, Avi; Karakos, Alexandros: Keystroke dynamics features for gender recognition. Digital Investigation, 24:4–10, 2018. DOI: https://doi.org/10.1016/j.diin.2018.01.018.

[TGG13] Tey, Chee Meng; Gupta, P.; Gao, D.: I can be You: Questioning the use of Keystroke Dynamics as Biometrics. NDSS, 2013.

[YCL06] Ylonen, T.; C. Lonvick, Ed.: The Secure Shell (SSH) Protocol Architecture. 01 2006. doi:10.17487/RFC4251. RFC 4251.

[YR06] Yao, Zizhen; Ruzzo, Walter L.: A Regression-based K nearest neighbor algorithm for gene function prediction from heterogeneous data. BMC Bioinformatics, 7(1):11, 03 2006. ISBN: 1471-2105, DOI: 10.1186/1471-2105-7-S1-S11.

[Za17] Zaidan, D.; Salem, A.; Swidan, A.; Saifan, R.: Factors affecting keystroke dynamics for verification data collecting and analysis. 2017 8th International Conference on Information Technology (ICIT), pp. 392–398, 2017. DOI: 10.1109/ICITECH.2017.8080032.