Int. J. Man-Machine Studies (1985) **23,** 263–273

Identity verification through keyboard characteristics

DAVID UMPHRESS AND GLEN WILLIAMS

Department of Computer Science, Texas A & M University, College Station,
Texas 77843-3131, U.S.A.

(*Received 15 January 1985 and in revised form 25 April 1985*)

Most personal identity mechanisms in use today are artificial. They require specific actions on the part of the user, many of which are not "friendly". Ideally, a typist should be able to approach a computer terminal, begin typing, and be identified from keystroke characteristics. Individuals exhibit characteristic cognitive properties when interacting with the computer through a keyboard. By examining the properties of keying patterns, statistics can be compiled that uniquely describe the user. Initially, a reference profile is built to serve as a basis of comparison for future typing samples. The profile consists of the average time interval between keystrokes (mean keystroke latency) as well as a collection of the average times required to strike any two successive keys on the keyboard. Typing samples are scored against the reference profile and a score is calculated assessing the confidence that the same individual typed both the sample and the reference profile. This mechanism has the capability of providing identity surveillance throughout the entire time at the keyboard.

Introduction

With increasing frequency, newspaper headlines are emphasizing the vulnerability of computer security measures. Numerous stories detail the break-in of computer systems, the perusal of classified files, and the destruction of invaluable information. Reading such stories should give the computer-professional pause to consider how system resources are protected. In examining methods by which users gain access to computer resources, the first items that come to mind are passwords, keys, badges, fingerprints and signatures. Even at this superficial level, it is not difficult to see that security mechanisms can be grouped into three general categories (Wood, 1978):

(1) what the user knows, e.g. passwords;
(2) what the user has, e.g. keys, badges;
(3) what the user is, e.g. fingerprints, signatures.

The degree of confidence with which each category protects computer resources lies in how readily the security mechanism can be circumvented. Each category above has advantages in terms of cost, ease of implementation, and user convenience. However, it is the third category, "what a user is", that presents the strongest line of defence against counterfeit users.

Close examination of most computer security systems reveals that they are not personal identifiers, but identity verifiers; that is, they do not require a multitude of identity measurements to produce a composite confidence of user identity. Instead, they require a single source of identity and verify authorization based on that source alone. The password is by far the most popular identity verifier due to its economical viability and ease of implementation: it requires no special hardware. Further, it is

0020-7373/85/090263+11$03.00/0

intangible, and therefore easy to transport from terminal to terminal and physically difficult to lose. However, the very intangibility of the password gives it the distinct disadvantage of being compromised without consent or knowledge of its disclosure. Moreover, it does not provide protection past the initial recognition stage.

The widespread use of the personal computer further compounds the problem of sole-source identity verification. When furnished with even the simplest of telecommunications equipment, the user of the home computer may initiate an automated search for computer dial-up ports. Once a dial-up port has been found, exhaustive attempts may be made to determine a valid password and thus allow the user to impersonate an authorized user. After circumventing the initial security measures, the imposter is free to browse files and scavenge information (Steinauer, 1981). In essence, the wide proliferation of the personal computer has permitted abusers to pick the lock of many data-processing installations.

The havoc unleashed by the home computer can be turned to an advantage. This paper examines a means of supplementing identity verification using the equipment that is available on existing personal computers. Further, the concept described not only adds to initial identity verification, but provides a means of constant identity surveillance.

Predictable human characteristics

As early as the turn of the century, psychology experiments demonstrated that the mechanics of human actions are predictable in the performance of repetitive, routine tasks. In 1895, observation of telegraph operators showed that each operator had a distinctive pattern of keying messages over telegraph lines (Bryan & Harter, 1973). Moreover, operators often recognized who was transmitting information simply by listening to the characteristic pattern of dots and dashes.

Just as the telegraph key served as a common input medium in days past, keyboards, light pens, joysticks and mice are common input devices today. The question posed now is one of whether properties exhibited in the use of these devices are unique to the individual user. Keyboard characteristics are rich in cognitive qualities and give great promise as a personal identifier. Anyone who sits within earshot of a typist or has an office next to a keypunch room is usually able to recognize typists by keystroke patterns. This paper presents a study of keystroke patterns as a supplement to identity verification. Part 1 establishes the framework for using keystroke characteristics as an identifier. Part 2 describes the results of an actual experiment in personal recognition.

1. Model of human behaviour

Human nature dictates that an individual does not sit before a computer and deluge the keyboard with a furious and continuous stream of nonstop data. Instead, the user types for a while, pauses to collect thoughts, types a bit more, pauses again to decide a new strategy, continues typing, and so forth. In developing a scheme for identity verification, a common base must be established for determining which keystrokes characterize the individual's key patterns and which do not. Psychological models describing the human interface with computer programs aid in this process. Several models are proposed in the literature. The most popular, the keystroke-level model, was chosen as a basis for this work (Card, Moran & Newell, 1980). It describes man-machine interaction during a session at a computer terminal. The model was intended as a tool for the evaluation and comparison of designs for highly interactive programs. Given this scope, however, it provides an interesting insight into human performance and, more importantly, human predictability.

The keystroke-level model summarizes the entire terminal session as:

$$T_{\text{task}} = T_{\text{acquire}} + T_{\text{execute}}$$

T_{task} represents the duration of the terminal session; T_{acquire} is the time required to assess the task, build a mental representation of the functions to be performed and choose a method for solving the problem and T_{execute} is the time required to call on the system resources to perform the tasks.

As one would expect, T_{acquire} varies according to the magnitude of the task at hand, the experience of the user and understanding of the functions to be performed. It is not quantifiable, therefore, and cannot be used to characterize individuals. T_{execute}, on the other hand, describes mechanical actions. It may be described further as:

$$T_{\text{execute}} = T_k + T_m \dagger$$

where T_k is the time to key in information and T_m is the time required for mental preparation. The T_m here may be thought of as tactical planning in contrast to T_{acquire} which is strategic in nature.

Such a macro view of T_{execute} does not depict the true processes that are occurring. When interacting with a program, the user does not separate his or her actions into mental time followed by keystroke time. Instead, the two are intermixed. Looking closer, T_{execute} can be portrayed as a series of mental/keystroke clusters as:

$$T_{\text{execute}} = \Sigma(T_{mi} + T_{ki}).$$

The expression in parentheses describes the fundamental human action of breaking a larger task into smaller, more easily managed, subtasks. Each subtask is known as a cognitive unit, or in more vernacular terms, a "chunk".

The representation of a data item being keyed into the computer can be seen with the following example. Suppose a user wishes to display the directory of a disk. Further, suppose that the command needed to accomplish this is, say DIR. The keystroke-level model would represent the actions required as:

MK[D]K[I]K[R]K[RETURN].

M constitutes T_m, the time required to conceptualize which keys must be activated in order to display the directory. The collection of Ks make up the actual keystrokes, the corresponding keystroke is enclosed in brackets.

Extrapolating from this, single commands can be linked serially to form entire typed lines. The authors of the model give elaborate heuristics for determining the placement of the M operator within the keyed input in an effort to show cognitive patterning. Intuitively, it is expected that those keystrokes within each cognitive unit are chosen as being characteristic of an individual's typing patterns. Examining keystroke patterns that span cognitive boundaries introduces the complicating factor of pauses caused by

† The keystroke-level model includes operators for time required to point a mouse, time required to draw lines, etc. These operators are independent of keystroke time and hence not included for discussion.

mental preparation time. This is a factor that is not necessarily quantifiable. Instead, keystrokes between M operators are most representative of individual key patterns.

Close examination reveals that even at the granularity of the cognitive unit identified by the model, certain keystrokes must be filtered further. Research has shown that when a typist is keying data, the brain acts as a buffer. The typist first looks at the text to be typed, loads a certain amount of text into the buffer, then outputs the text onto the keys of the keyboard. The buffer is, on the average, 6–8 characters in length (Shaffer, 1973). Due to this limitation in buffer size, typists group symbols into smaller cognitive units and pause between each unit. Typical pause points are between words as well as within words that are longer than 6–8 characters (Cooper, 1983). In light of this, the definition of a predictable cognitive unit must be restricted. Only keystroke patterns within the first 6–8 characters of words will be considered as candidates for characterization.

2. Experiment in keystroke characterization

The psychological model describes which keystrokes provide meaningful information towards an individual's particular key pattern. The problem now becomes one of collecting statistics that characterize the key pattern. Two sets of inputs are required for user identification, a reference profile and a test profile.

REFERENCE PROFILE

The reference profile serves as a control; it is the basis for all subsequent comparisons to determine personal identification. To build a reference profile, an individual takes a standard typing test. Each keystroke is time-tagged and stored for analysis. The participant is instructed not to attempt to correct typing errors. Typists often realize they have made a mistake immediately after the error has been made (Shaffer, 1970). In the process of catching the mistake, the typist unconsciously pauses before completing the remainder of the word. This introduces an extraneous cognitive boundary that misrepresents the normal keystroke pattern. For this reason, the first step in the filtering process is to compare the test keystrokes to the test text. Words containing errors are discarded. This is done in an effort to obtain as clean a reference profile as possible.

After screening the entered text for errors, the keystrokes are grouped into words. Keystroke latencies are calculated by taking the difference between the times of each pair of adjacent keystrokes. Latencies are calculated only for the first six keystrokes in each word. If the word is longer than six characters, the remaining keystrokes are ignored.

The final steps of the elimination process attempt to compensate for possible anomalous keystroke latencies. First, latencies over 0·75 s in duration are discarded. Latencies of over 0·75 s indicate that the typist is unfamiliar with the keyboard (Fig. 1). Thus, that particular keystroke is not a good candidate for inclusion in the typist's keystroke profile. Second, the latency time for capital letters is halved to allow for the two keystrokes required to form capital letters. Since latency times are considered only for keystrokes within words, a place where capital letters seldom appear, this has little effect on the pattern recognition process.

Two measures of key patterning are produced from the filtered keystrokes. The first measure is the mean and standard deviation keystroke latency. The second indicator

Skill level	Keystroke speed (wpm)	Keystroke latency
Best	135	0·08
Good	90	0·12
Average skilled	55	0·20
Average non-skilled	40	0·28
Poor	25	0·48
Unfamiliar with keyboard		1·20

FIG. 1. Comparison of keystroke speeds (Card *et al.*, 1980).

describes the latency between all adjacent letter combinations by defining a 26×26 matrix, whose rows correspond to the first letter of a two letter digraph, and whose columns correspond to the second letter. Each cell in the matrix gives the average keystroke latency of the diagraph defined by the cell position. Fig. 2 summarizes the process used to form the reference profile.

TEST PROFILE

The test profile is the collection of keystrokes produced by an individual requesting identity verification. The crux of the problem here is to compare the test profile with a reference profile and assign a confidence that the individual typing the test profile is the same as the one who typed the reference profile. Two methods of comparison are possible. One approach treats the test profile in much the same way as the reference profile was analysed. Here, keystrokes are collected, time-tagged and screened for errors. Spurious latencies are eliminated, statistics are computed and a matrix of digraph latencies is constructed. The statistics and matrix are then compared with the reference profile at one time in a batch-type manner. The second comparison method evaluates keystrokes in real-time.

Since an implementation strategy was devised based on real-time performance aspects, the second method will be discussed in depth. Using this method, keystrokes of the test profile are time-tagged in the same manner as in the reference profile. After each key is depressed, the difference between the time from the previous keystroke is calculated in real time. Keystrokes are filtered according to the cognitive principles discussed for the reference profile. First, only keystroke latencies within words are considered for comparison. That is, if any keystroke of the current digraph is other than an alphabetic character, the latency time is ignored. Second, if the second character is a backspace, an error is assumed to have occurred within the word. This and all subsequent keystroke latencies within the word are ignored. Third, only latencies between the first six characters of words are considered. Finally, latencies over 0·75 s in duration are discarded.

SCORING

Two tests are performed to determine how closely the test profile matches the reference profile. The first test assesses keystroke intervals within character patterns and the second test appraises overall typing characteristics.

For the first test, the keystroke latency is compared to the appropriate cell in the digraph matrix of the reference profile. The cell position is determined by using the

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE

fastcase®
Smarter legal research.