Direct and Indirect Human Computer Interaction Based Biometrics

Roman V. Yampolskiy
University at Buffalo, Buffalo, USA
rvy@buffalo.edu

Venu Govindaraju
University at Buffalo, Buffalo, USA
govind@buffalo.edu

Abstract—**In this paper we survey the state of the art in direct and indirect human computer interaction based biometrics. Direct HCI biometrics are based on abilities, style, preference, knowledge, or strategy used by people while working with a computer. The indirect HCI-based biometrics are events that can be obtained by monitoring users' HCI behavior indirectly via observable low-level actions of computer software. We examine current research and analyze the types of features used to describe HCI behavior. After comparing accuracy rates for verification of users using different HCI-based biometric approaches we address privacy issues which arise with the use of HCI dependant biometrics. Finally, we present results of our experiments with direct and indirect HCI-based behavioral biometrics employed as a part of an intrusion detection system.**

Index Terms—**behavioral biometrics, human computer interaction, intrusion detection.**

I. Introduction

With the proliferation of computers and of the Internet in our every day lives need for reliable computer security steadily increases. Research in biometric technologies offers one of the most promising approaches to providing user friendly and reliable control methodology for access to computer systems and networks. Majority of such research is aimed at studying well established physical biometrics such as fingerprint or iris scans [1]. Human Computer Interaction (HCI) explores how human beings interact with computational devices. This type of interaction, relatively unique to every computer user can be analyzed to develop a non-intrusive authentication mechanism. HCI-based biometrics are usually only briefly mentioned and only those which are in large part based on muscle control such as keystrokes, or mouse dynamics are well researched [2]. In this paper we concentrate on reviewing and analyzing all existing HCI-based biometric technologies.

HCI-based biometrics provide a number of advantages over traditional biometric technologies. They can be collected non-obtrusively or even without the knowledge of the user. Collection of data usually does not require any special hardware and is so very cost effective. While HCI-based biometrics are not unique enough to provide reliable human identification they have been shown to provide high accuracy identity verification.

In their interaction with computers human beings employ different strategies, use different style and apply unique abilities and knowledge. Intrusion detection researchers attempt to quantify such HCI-based-biometric traits and use resulting feature profiles to successfully verify user identity and reject intruders. HCI-based biometrics can be subdivided into two different categories known as direct and indirect HCI-based biometrics [3].

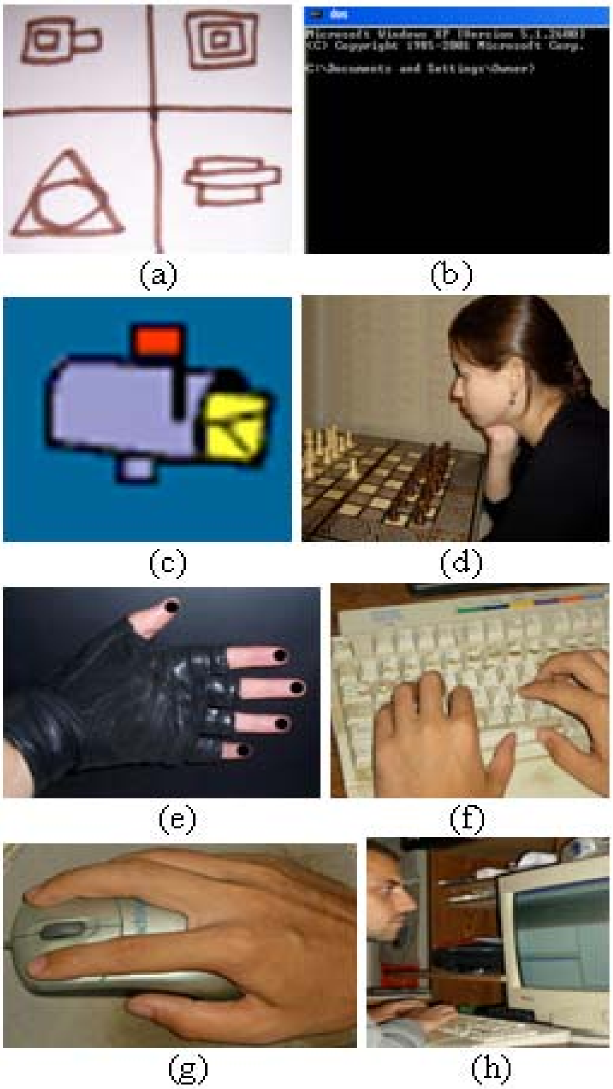
First group is made up of those biometrics which are based on direct human interaction with input devices such as keyboard [4-10], computer mouse [11-15], and haptics [16-18] which rely on supposedly innate, unique and stable muscle actions [19] and those biometrics which are based on advanced human behavior such as strategy, knowledge or skill exhibited by the user during interaction with different software [3]. Examples of such high level HCI-based behavioral biometrics include: email behavior [20, 21], programming style [22-24], utilized online game strategy [25-27], biometric sketch [28, 29], and command line lexicon [30-33].

The second group consists of the indirect HCI-based biometrics which are events that can be obtained by monitoring user's HCI behavior indirectly via observable low-level actions of computer software, those include audit logs, call-stack data, GUI interaction, network traffic, registry access, storage activity, and system calls [11, 34]. These low-level events are produced unintentionally by the user during interaction with different software applications during pursuit of some, potentially mischievous, high level goals.

Based on "Human Computer Interaction Based Intrusion Detection", by Roman V. Yampolskiy, which appeared in the Proceedings of the IEEE International Conference on Information Technology: New Generations (ITNG2007), Las Vegas, Nevada, USA, April 2-4, 2007. © 2007 IEEE.

II. Direct HCI-Based Biometrics

In this section we present an overview of the most established Direct Human Computer Interaction-Based Biometrics (DHCIBB). DHCIBB can be subdivided into two different categories, first one consisting of human interaction with input devices such as keyboards, mice, and haptics which rely on supposedly innate, unique and stable muscle actions [19]. The second group consists of HCI-based behavioral biometrics which measure advanced human behavior such as strategy, knowledge or skill exhibited by the user during interaction with different software. Figure 1 shows some of the most popular examples of the DHCIBB. Additional methods exist that rely on monitoring a user's HCI behavior indirectly [11], those include monitoring system call traces [35], audit logs [36], program execution traces [37], registry access [38], storage activity [39], and call-stack data analysis [40], but those approaches are beyond the scope of this paper.



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Owner>
```

Figure 1. Examples of Direct HCI-Based Behavioral Biometrics: a) Biometric Sketch, b) Command Line Lexicon, c) Email, d) Game Strategy, e) Haptic, f) Keystrokes, g) Mouse Dynamics, h) Programming Style.

A. Input Device Interaction Based Biometrics

1) Keystroke Dynamics: Typing patterns are characteristic to each person, some people are experienced typists utilizing the touch-typing method, and others utilize the hunt-and-peck approach which uses only two fingers. Those differences make verification of people based on their typing patterns a proven possibility, some reports suggest identification is also possible [4]. For verification a small typing sample such as the input of user's password is sufficient, but for recognition a large amount of keystroke data is needed and identification is based on comparisons with the profiles of all other existing users already in the system.

Keystroke features are based on time durations between the keystrokes, inter-key strokes and dwell times, which is the time a key is pressed down, overall typing speed, frequency of errors (use of backspace), use of numpad, order in which user presses shift key to get capital letters and possibly the force with which keys are hit for specially equipped keyboards [4, 5]. Keystroke dynamics is probably the most researched type of HCI-based biometric [6, 7], with novel research taking place in different languages [8], for long text samples, [9, 10] and for email authorship identification [41].

2) Mouse Dynamics: By monitoring all mouse actions produced by the user during interaction with the Graphical User Interface (GUI), a unique profile can be generated which can be used for user re-authentication [11]. Mouse actions of interest include general movement, drag and drop, point and click, and stillness. From those a set of features can be extracted for example average speed against the distance traveled, and average speed against the movement direction [12, 13]. Pusara et al. [11] describe a feature extraction approach in which they split the mouse event data into mouse wheel movements, clicks, menu and toolbar clicks. Click data is further subdivided into single and double click data.

Gamboa et al. [14, 15] have tried to improve accuracy of mouse-dynamics-based biometrics by restricting the domain of data collection to an online game instead of a more general GUI environment. As a result applicability of their results is somewhat restricted and the methodology is more intrusive to the user. The system requires around 10-15 minutes of devoted game play instead of seamless data collection during the normal to the user human computer interaction. As far as the extracted features, *x* and *y* coordinates of the mouse, horizontal velocity, vertical velocity, tangential velocity, tangential acceleration, tangential jerk and angular velocity are utilized with respect to the mouse strokes to create a unique user profile.

*3) Haptic***:** Haptic systems are computer input/output devices which can provide us with information about direction, pressure, force, angle, speed, and position of user's interactions [16, 17]. Because so much information is available about the user's performance a high degree of accuracy can be expected from a haptic based biometrics system. Orozco et al. [16, 17] have created a simple haptic application built on an elastic membrane surface in which the user is required to navigate a stylus through the maze. The maze has gummy walls and a stretchy floor. The application collects data about the ability of the user to navigate the maze, such as reaction time to release from sticky wall, the route, the velocity, and the pressure applied to the floor. The individual user profiles are made up of such information as 3D world location of the pen, average speed, mean velocity, mean standard deviation, navigation style, angular turns and rounded turns.

In a separate experiment Orozco et al. [18] implement a virtual mobile phone application where the user interacts through a haptic pen to simulate making a phone call via a touch pad. The keystroke duration, pen's position, and exerted force are used as the raw features collected for user profiling.

B. Software Interaction Based Biometrics

1) Email Behavior: Email sending behavior is not the same for all individuals. Some people work at night and send dozens of emails to many different addresses; others only check mail in the morning and only correspond with one or two people. All this peculiarities can be used to create a behavioral profile which can serve as a behavioral biometric for an individual. Length of the emails, time of the day the mail is sent, how frequently inbox is emptied and of course recipients' addresses among other variables can all be combined to create a baseline feature vector for the person's email behavior. Some work in using email behavior modeling was done by Stolfo et al. [20, 21]. They have investigated possibility of detecting virus propagation via email by observing abnormalities in the email sending behavior, such as unusual clique of recipients for the same email. For example sending the same email to your girlfriend and your boss is not an everyday occurrence.

De Vel et al. [42] have applied authorship identification techniques to determine the likely author of an email message. Alongside the typical features used in text authorship identification such as count of function-words and word length frequency distribution authors also used some email specific structural features such as: use of a greeting, farewell acknowledgment, signature, number of attachments, position of re-quoted text within the message body, HTML tag frequency distribution and total number of HTML tags. Overall, almost 200 features are used in the experiment, but some frequently cited features used in text authorship determination are not appropriate in the domain of email messages due to the shorter average size of such communications.

2) Programming Style: With the increasing number of viruses, worms, and Trojan horses it is often useful in a forensic investigation to be able to identify an author of such malware programs based on the analysis of the source code. It is also valuable for the purposes of software debugging and maintenance to know who the original author of a certain code fragment was. Spafford et al. [22] have analyzed a number of features potentially useful for the identification of software authorship. In case only the executable code is available for analysis, data structures and applied algorithms can be profiled as well as any remaining compiler and system information, observed programming skill level, knowledge of the operating system and choice of the system calls. Additionally use of predefined functions and provisions for error handling is not the same for different programmers.

In case the original source files are available a large number of additional identifying features become accessible such as: chosen programming language, code formatting style, type of code editor, special macros, comment style, variable names, spelling and grammar, use of language features such as choice of loop structures, the ratio of global to local variables, temporary coding structures, and finally types of mistakes observable in the code. Software metrics such as number of lines of code per function, comment-to-code ratio and function complexity may also be introduced [22]. Similar code features are discussed by Gray et al. [23] and in Grantzeskou et al. [24].

3) Computer Game Strategy: Ramon et al. [27] have demonstrated possibility of identifying Go players based on their style of game play. They analyzed a number of Go specific features such as type of opening moves, how early such moves are made and total number of liberties in the formed groups. They also speculate that the decision tree approach they have developed can be applied to other games such as Chess or Checkers.

Jansen et al. [43] report on their research in chess strategy inference from game records. In particular they were able to surmise good estimates of the weights used in the evaluation function of computer chess players and later applied same techniques to human grandmasters. Their approach is aimed at predicting future moves made by the players, but the opponent model created with some additional processing can be utilized for opponent identification or at least verification. This can be achieved by comparing new moves made by the player with predicted ones from models for different players and using the achieved accuracy scores as an indication of which profile models which player.

4) Biometric Sketch: Bromme et al. [28, 29] proposed a biometric sketch authentication method based on sketch recognition and a user's personal knowledge about the drawings content. The system directs a user to create a simple sketch for example of three circles and each user is free to do so in any way he pleases. Because a large number of different combinations exist for combining multiple simple structural shapes sketches of different users are sufficiently unique to provide accurate authentication. The approach measures users' knowledge about the sketch, which is only available to the previously authenticated user. Such features as the sketches location and relative position of different primitives are taken as the profile of the sketch. Similar approaches are tried by Varenhorst [44] with a system called Passdoodles and also by Jermyn et al. [45] with a system called Draw-a-Secret. Finally a V-go Password requests a user to perform simulation of simple actions such as mixing a cocktail using a graphical interface, with the assumption that all users have their unique approach to bartending [46].

5) Command Line Lexicon: A popular approach to the construction of behavior based intrusion detection systems, is based on profiling the set of commands utilized by the user in the process of interaction with the operating system. A frequent target of such research is UNIX operating system, probably due to it having mostly command line nature. Users differ greatly in their level of familiarity with the command set and all the possible arguments which can be applied to individual commands. Regardless of how well a user knows the set of available commands; most are fairly consistent in their choice of commands used to accomplish a particular task.

A user profile typically consists of a list of used commands together with corresponding frequency counts, and lists of arguments to the commands. Data collection process is often time consuming since as many as 15,000 individual commands need to be collected for the system to achieve high degree of accuracy [47, 48]. Additional information about the secession may also be included in the profile such as the login host and login time, which help to improve accuracy of the user profile as it is likely that users perform different actions on different hosts [49]. Overall, this line of research is extremely popular [30-33], but recently a shift has been made towards user profiling in a graphical environment such as Windows as most users prefer convenience of a Graphical User Interface (GUI). Typical features extracted from the user's interaction with a windows based machine include: time between windows, time between new windows, number of windows simultaneously open, and number of words in a window title. [50, 51].

III. Indirect HCI-Based Biometrics

Indirect HCI-based biometrics are sometimes known to different researchers under different names. IDS based on system calls or audit logs are often classified as utilizing program execution traces and those based on call-stack data as based on system calls. The confusion is probably caused by the fact that a lot of interdependency exists between different indirect behavioral biometrics and they are frequently used in combinations to improve accuracy of IDS being developed. For example system calls and program counter data may be combined in the same behavioral signature or audit logs may contain information about system calls. Also we can't forget that a human intruder is indirectly behind each one of those reflections of behavior and so a large degree of correlation is to be expected. In this section we tried to distill all indirect HCI-based biometrics into the seven well defined groups, but some overlay undoubtedly exists [52].

A. Audit Logs

Most modern operating systems keep some records of user activity and program interaction. While such audit trails can be of some interest to intrusion detection researchers, specialized audit trails specifically designed for security enforcement can be potentially much more powerful. A typical audit log may contain such information as: CPU and I/O usage, number of connections from each location, whether a directory was accessed, a file created, another user ID changed, audit record was modified, amount of activity for the system, network and host [53]. Experimentally it has been shown that collecting audit events is a less intrusive technique than recording system calls [54]. Because an enormous amount of auditing data can be generated overwhelming an intrusion detection system it has been suggested that a random sampling might be a reasonable approach to auditing data [55]. Additional data might be helpful in distinguishing suspicious activity from normal behavior. For example facts about changes in user status, new users being added, terminated users, users on vocations, or changed job assignments might be needed to reduce the number of false positives produced by the IDS [53]. Since so much potentially valuable information can be captured by the audit logs a large number of IDS researchers are attracted to this form of indirect HCI-based biometric [35, 36, 56-62].

B. Call-Stack

Feng et al. [63] developed a method for performing anomaly detection using call stack information. The program counter indicates the current execution point of a program; and since each instruction of a program corresponds to a unique program counter this information may be useful for intrusion detection. The idea is to extract return addresses from the call stack and generate an abstract execution path between two program execution points. This path is analyzed to decide whether this path is valid based on what has been learned during the normal execution of the program. Return addresses are a particularly good source of information on suspicious behavior. The approach has been shown capable of detecting some attacks that could not be detected by other approaches, while retaining a comparable false positive rate [63]. Additional research into call-stack-based intrusion detection has been performed by Giffin et al. [64].

C. GUI Interaction

Expanding on the idea of monitoring user's keyboard and mouse activity Garg et al. [34] developed a system for collecting Graphical User Interface (GUI) interaction-based data. Collected data allows for generation of advanced behavioral profiles of the system's users. Such comprehensive data may provide additional information not available form typically analyzed command line data. With proliferation of GUI based systems a shift towards IDS based on GUI interaction data, as opposed to command line data, is a natural progression. Ideally the collected data would include high-level detailed information about the GUI related actions of the user such as: left click on the Start menu, double click on explorer.exe, close Notepad.exe window, etc. Software generated by Garg et al. records all possible low-level user activities on the system in real time, including: system background processes, user run commands, keyboard activity and mouse clicks. All collected information is time stamped and preprocessed to reduce the amount of data actually used for intrusion detection purposes [34].

D. Network Traffic

Network level intrusion detection is somewhat different from other types of intrusion detection as the monitored activity originates outside the system being protected. With the increase in popularity of Internet and other networks an intruder no longer has to have physical access to the system he is trying to penetrate. This means that the network dataflow arriving on different system ports and encoded using different protocols needs to be processed and reviewed. IDS based on network traffic analyze various packet attributes such as: IP protocol-type values, packet size, server port numbers, source and destination IP prefixes, Time-To-Live values, IP/TCP header length, incorrect IP/TCP/UDP checksums, and TCP flag patterns. During the baseline profiling period the number of packets with each attribute value is counted and taken as normal behavior [65]. Any deviation from the normal baseline profile may set an alert flag informing network administrator that an attack is taking place. Many IDS have been developed based on the concept of network level attack detection [66-71] and the general area of network traffic analysis is highly applicable for improved network and network application design [72, 73].

E. Registry Access

Apap et al. [38] proposed a new host IDS they call Registry Anomaly Detection (RAD) that monitors access to the Windows registry in real time and detects the actions of malicious software. Windows registry stores information about hardware installed on the system, which ports are used, user profiles, policies, user names, passwords and configuration settings for programs. Most programs access a certain set of registry keys during normal operation. Similarly most users use only a certain subset of programs available on the machine. This results in a high degree of regularity in registry interaction during the normal operation of the system. However, malicious software may substantially deviate from this regular activity and can be detected. Many attacks involve starting programs which have rarely been used in the past or changing keys that have never been changed before. If a RAD system is trained on clean data, then these kinds of registry operations will appear abnormal to the system and result in issue of an alert [38].

F. Storage Activity

Many actions of intruders became visible at the storage level interface. Manipulation of system utilities (to add backdoors), tampering with audit logs (to destroy evidence), resetting of attributes (to hide changes) and addition of suspicious content (known virus) all show up as the changes in the storage layer of the system. A storage-based IDS analyzes all requests received by the storage server and can issue alerts about suspicious activity to the system administrator. Additionally it can slow down the suspected intruder's storage access or isolate intruder via a forking of version trees to a sandbox. Storage-based IDS has the advantage of being independent from the client's operating system and so can continue working after the initial compromise, unlike host-based IDS which can be disabled by the intruder [39]. Research using storage activity is fast gaining in popularity with intrusions being detected at the block storage level and in Storage Area Networks (SAN) environments [74], object-based storage devices [75], and workstation disk drives [76].

G. System Calls

A system call is the method used by a program to request service from the operating system, or more particularly, the operating system kernel. System calls use a special instruction which causes the processor to transfer control to a more privileged code segment. Intruder detection can be achieved by comparing an application's run-time system calls with a pre-defined normal system call behavior model. The assumption is that as long as the intruder can't make arbitrary system calls, it is unlikely that he can achieve his desired malicious goals [77]. Following the original work of Forest et al. [78, 79] a number of researchers have pursuit development of IDS based on analyzing system call sequences [64, 77, 80-83]. Typically a model of normal system call behavior is learned during the training phase which is a baseline-state assumed to be free of attacks [84], alternative approaches use static analysis of the source code or binary code [64]. A number of representation schemas for the behavioral model have been proposed, including strings [54, 79], finite state automata and push down automata [63, 64].

IV. Comparison and Analysis

All of the presented direct HCI-based biometrics share a number of characteristics and so can be analyzed as a group using seven properties of good biometrics presented by Jain et al. [1, 5].

- **Universality** HCI-based biometrics are dependent on specific abilities possessed by different people to a different degree or not at all and so in a general population universality of HCI-based biometrics is very low. But since HCI-based biometrics are only applied to those who participate in computer interactions, actual universality of HCI-based biometrics is a 100%.
- **Uniqueness** Since only a small set of different approaches to performing any task on a computer exists uniqueness of HCI-based biometrics is relatively low. Number of existing programming

styles, different online game strategies and varying preferences are only sufficient for user verification not identification unless the set of users is extremely small.

- **Permanence** HCI-based biometrics exhibit a low degree of permanence as they measure behavior which changes with time as person learns advanced techniques and faster ways of accomplishing tasks. However, this problem of concept drift is addressed in the behavior based intrusion detection research and systems are developed capable of adjusting to the changing behavior of the users [85, 86].
- **Collectability** Collecting HCI-based biometrics is relatively easy and unobtrusive to the user. In some instances the user may not even be aware that data collection is taking place. The process of data collection is fully automated and is very low cost.
- **Performance** The identification accuracy of HCI-based biometrics is very low particularly as the number of users in the database becomes large. However verification accuracy can be very good for some HCI-based biometrics.
- **Acceptability** Since HCI-based biometrics can be collected without user participation they enjoy a high degree of acceptability, but might be objected to for ethical or privacy reasons.
- **Circumvention** It is relatively difficult to get around HCI-based biometric systems as it requires intimate knowledge of someone else's behavior, but once such knowledge is available fabrication might be very straightforward. This is why it is extremely important to keep the collected user profiles securely encrypted.

TABLE I
DIRECT HCI-BASED BIOMETRICS TRAITS AND FEATURES

HCI-Based Biometric	**Measures**	**Extracted Features**
Keystroke Dynamics	Skill	time durations between the keystrokes, inter-key strokes and dwell times, which is the time a key is pressed down, overall typing speed, frequency of errors (use of backspace), use of numpad, order in which user presses shift key to get capital letters
Email Behavior	Style	length of the emails, time of the day the mail is sent, how frequently inbox is emptied, the recipients' addresses
Mouse Dynamics	Style	*x* and *y* coordinates of the mouse, horizontal velocity, vertical velocity, tangential velocity, tangential acceleration, tangential jerk and angular velocity
Program-ming Style	Skill, Style, Preferences	chosen programming language, code formatting style, type of code editor, special macros, comment style, variable names, spelling and grammar, language features, the ratio of global to local variables, temporary coding structures
Haptic	Style	3D world location of the pen, average speed, mean velocity, mean standard deviation
Online Game Strategy	Strategy/ Skill	count of hands folded, checked, called, raised
Biometric Sketch	Knowledge	location and relative position of different primitives
Command Lexicon	Technical Vocabulary	used commands together with corresponding frequency counts

All direct HCI-based biometrics essentially measure human actions which result from unique to every human abilities, style, preference, knowledge, or strategy. Table I summarizes what precisely is being measured by different direct HCI-based biometrics as well as lists some of the most frequently used features for each biometric approach.

While many HCI-based biometrics are still in their infancy some very promising research has already been done. The results obtained justify feasibility of using human computer interaction for verification of individuals and further research in this direction is likely to improve accuracy of such systems. Table II summarizes accuracy rates obtained by different researchers.

TABLE II
REPORTED VERIFICATION ACCURACY (DIRECT)

Behavioral Biometric	**Publication**	**Detection Rate**	**FAR**	**FRR**	**EER**
Biometric Sketch	Bromme 2003 [28]				7.2%
Command Lexicon	Marin 2001 [31]	74.4%		33.5%	
Email Behavior	de Vel 2001 [42]	90.5%			
Game Strategy	Yampolskiy 2007 [87]				7.0%
Haptic	Orozco 2006 [17]		25%		22.3%
Keystroke Dynamics	Bergadano 2002 [7]		0.01%	4%	
Mouse Dynamics	Pusara 2004 [11]		0.43%	1.75%	
Programming Style	Frantzeskou 2004 [24]	73%			

Table III summarizes detection rates obtained by different researchers while utilizing indirect HCI-based biometrics.

TABLE III
DETECTION AND FALSE POSITIVE RATES (INDIRECT)

Type of Indirect Biometric	**Publication**	**Detection Rate**	**False Positive Rate**
Audit Logs	Lee 1999 [59]	93%	8%
Call-Stack	Feng 2003 [40]	-	1%
GUI Interaction	Garg 2006 [34]	96.15%	3.85%
Network Traffic	Zhang 2003 [67]	96.2%	.0393%
Registry Access	Apap 2001 [38]	86.9%	3.8%
Storage Activity	Stanton 2005 [88]	97%	4%
System Calls	Ghosh 1999 [83]	86.4%	4.3%

V. Generalized Procedure and Ethical Issues

We propose a generalized algorithm for HCI-based biometrics, which can be applied to any type of human computer interaction. The first step is to break up the behavior in question into a number of atomic operations each one corresponding to a single action. Ideally all possible operations should be considered, but in a case of behavior with a very large repertoire of possible operations a large subset of most frequent operations might be sufficient [2].

User's behavior should be observed and a frequency count for the occurrence of the atomic operations should be produced. The resulting frequency counts form a feature vector which is used to verify or reject the user based on the similarity score produced by a similarity function. An experimentally determined threshold serves as a decision boundary for separating legitimate users from intruders. In case user identification is attempted a neural network or a decision tree approach might be used to select the best matching user from the database of existing templates. Below we outline the proposed generalized algorithm.

1. Pick a type of HCI behavior
2. Break up behavior into component actions
3. Determine frequencies of component actions for each user
4. Combine results into a feature vector profile
5. Apply similarity measure function to the stored template and current behavior
6. Experimentally determine a threshold value
7. Verify or reject user based on the similarity score comparison to the threshold value

Because direct HCI-based biometrics measure our personal traits any data collected in the process of generation of a user profile needs to be safely stored in an encrypted form. An additional property of HCI-based profiles is that they might contain information which might be of interest to third parties which might discriminate against individuals based on such information. As a consequence intentionally revealing or obtaining somebody else's biometric profile for the purposes other than verification would be highly unethical. Examples of private information which might be revealed by some direct HCI-based biometric profiles follow:

- **Keystroke Dynamics** May indicate that an individual is physically challenged or less seriously has a poor typing techniques and is so an inefficient employee.
- **Haptics** Similarly to keystroke dynamics and mouse dynamics may reveal motor control problems of a particular user.
- **Email Behavior** An employer would be interested to know if employees send out personal emails during office hours.
- **Programming Style** Software metric obtained from analysis of code may indicate a poorly performing coder and as a result jeopardize the person's employment.
- **Computer Game Strategy** If information about game strategy is obtained by the player's opponents it might be analyzed to find weaknesses in the player's game and as a result be financially costly.
- **Command Line Lexicon** Information about proficiency with the commands might be used by an employer to decide if you are sufficiently qualified for a job involving computer interaction.

VI. Direct HCI-Based Experiments

We have developed a methodology for treating the strategy used while playing an online game as a type of a behavioral biometric. The game of poker was used as an example of a game with a clearly identifiable player strategy. A profile signature produced for each player was used as the person's behavioral biometric profile. This approach can be utilized by online casinos to detect a hacker who is using a stolen account. This is currently a major problem in the world of online game networks and a successful solution can be beneficial not just from theoretical but also from a practical point of view.

First a user profile is generated either by data mining an existing database of poker hands or by observing a live game of poker. To study the strategy of poker players scientifically, we needed to quantify and statistically analyze their behavior. In order to do so we defined a number of variables associated with actions of poker players. The parameters chosen were selected because they can be easily tracked by relatively straightforward methodologies and more importantly they are believed to accurately describe the long-term model of player's behavior for poker [25, 26].

The profile consists of frequency measures indicating range of cards considered by the player at all stages of the game. It also measures how aggressive the player is via such variables as percentages of re-raised hands. The profile is actually human readable meaning that a poker expert can analyze and understand strategy employed by the player from observing his or her behavioral profile [89]. Table IV demonstrates a sample profile for a player named Bob.

TABLE IV
A SAMPLE STRATEGY BASED BEHAVIORAL PROFILE [26]

Player Name: Bob			Hands Dealt: 224	
	Pre-Flop	Flop	Turn	River
# of Hands	224	68	46	33
Folded	67%	28%	24%	18%
Checked	7%	54%	52%	52%
Called	21%	32%	28%	33%
Raised	4%	1%	4%	6%
Check-Raised	0%	4%	0%	0%
Re-Raised	0%	1%	0%	0%
All-In	1%	3%	4%	39%

In the Table IV we see a 24 dimensional feature vector (number of hands played is only used to determine if we have enough information to put confidence in our statistical profile and is not counted as a part of a profile) [26].

Next, a similarity measure is obtain between the feature vector generated based on the recently collected player data and the data for the same player obtained in previous sessions. A score is generated indicating how similar the current style of play is to the historically shown style of play for a particular player. If a score is above a certain threshold, it might indicate that a different user from the one who has originally registered is using the account and so the administrator of the network needs to be alerted to that fact. If the score is below some threshold, the system continues collecting and analyzing the player data. We used Euclidian Distance as a similarity measure in our implementation [25, 26].

For the user verification experiment a databank of 30 player signatures each one was compared with one profile taken from the same player as the one who generated the original signature and with another profile taken from a randomly chosen player. Giving us an experimental set up in which intruders and legitimate users are equal in number. Using our similarity measure and a threshold of 75 the algorithm has positively verified 46.66% (28) users. The False Accept Rate (FAR) was 13.33% (8 users) and False Reject Rate (FRR) was only 8.33% (5 users). This gives us player verification with overall 78.33% accuracy. The value of the threshold is not a universal constant and depends most of all on the maximum FRR and FAR, which can be tolerated by the application. Additional factors such as the similarity function used and the structure and size of the behavioral signature utilized also influence the choice of the optimal threshold value [25, 26].

VII. Indirect HCI-Biometrics Experiments

In biometric recognition a baseline profile for the user is obtained during the enrollment period. During the biometric system's operation newly submitted biometric samples are recorded and compared to the database of the user profiles obtained during the enrollment period. If a match above a certain threshold value is detected the system can recognize the user's identity. Similar signature based approach can be employed towards the development of an intrusion detection system based on indirect HCI-based biometrics. During the training phase the system learns particular signatures of different attacks, which during deployment it can identify and as a result recognize a particular attack is taking place.

In this section we demonstrate application of an indirect HCI-based biometric towards the goal of developing an intrusion detection system. We have selected network traffic dataflow as a representative example of an indirect HCI-based biometric. Any attempt by the attacker to probe or permeate a remote system will undoubtedly result in detectable changes to the network dataflow and as a result make the intrusion potentially detectable even in real time. We successfully use Artificial Neural Networks (ANN) namely Multiple Layer Perceptron (MLP) and Radial Basis Function (RBF) Network to accomplish attack behavior recognition. Achieved detection rates either exceed or match those shown by previously developed systems of other researchers on the same dataset.

A. Data

To conduct the experiments we used the data from benchmarks of the International Knowledge Discovery and Data Mining group (KDD). These data are based on the benchmark of the Defense Advanced Research Projects Agency (DARPA) that was collected by the Lincoln Laboratory of Massachusetts Institute of Technology in 1998 [90]. The data consists of a number of basic network dataflow features such as: duration of the connection, protocol type, such as TCP, UDP or ICMP, service type, such as FTP, HTTP, Telnet, status flag, total bytes sent to destination host, total bytes sent to source host, whether source and destination addresses are the same or not, number of wrong fragments, and number of urgent packets. Each record consists of 41 attributes and one target. The target value indicates the attack name [59, 69, 70, 91].

First, the dataset was split into multiple files and duplicate records were removed. Each file contained records corresponding to a certain attack or normal behavior. Thus, a library of attacks was created. It was done to achieve an efficient way to format, optimize, and compose custom training and testing datasets. Second, symbolic features like attack name (23 different symbols), protocol type (three different symbols), service (70 different symbols), and flag (11 different symbols) were mapped to integer values ranging from 0 to N-1 where N is the number of symbols. Third, a certain scaling had taken place: each of the mapped features was linearly scaled to the range [0.0, 1.0]. Features having integer value ranges like duration were also scaled linearly to the range of [0, 1]. All other features were either Boolean, like logged_in, having values (0 or 1), or continuous, like diff_srv_rate, in the range of [0, 1]. No scaling was necessary for these attributes. Attacks with the most number of records were chosen to be in the training set. The following attacks were used to train and to test the neural networks: Smuf, Satan, Neptune, Ipsweep, Back [69, 70].

B. Classifier

A typical RBF Network consists of three layers of neurons: input, hidden and output. Each neuron belonging to the hidden layer represents a cluster in the input data space. The hidden layer, as a whole, is a series of such clusters. A radial function, typically Gaussian, serves as an activator for each of the centers. The output for the activation function is determined based on the Euclidian distance between the center and the input vectors. The output neurons calculate a weighted sum of the hidden neurons. Input data values are uniquely assigned to the neurons in the input layer, which pass the data on to the hidden layer directly without any weights. Hidden layer neurons are called RBF units and are determined by a parameter vector called center and a scalar value called width [92]. RBF network had 41 inputs, corresponding to each attribute in the dataset, two outputs for attack detection (the first output for the presence of an attack – "YES", the second output for the normal behavior – "NO"), or six outputs for attack classification (five outputs for the attacks, and the sixth output for the normal behavior), three layers (input, hidden, and output). The training set consisted of 4000 records. The attack and the normal behavior records were evenly distributed in the training set [69, 70].

The MLP network consists of several layers of neurons. Each neuron in a certain layer is connected to each neuron of the next layer. There are no feedback connections. The weights are considered as NN parameters to be adjusted during training. The most often used MLP-network consists of three layers: an input layer, one hidden layer, and an output layer. The hidden and output layers usually have a non-linear activation function. Typically, some type of backpropagation algorithm is used in training this type of network [92]. The parameters of the MLP ANN were: 41 inputs, corresponding to each attribute in the dataset. Two outputs for attack detection (the first output for the presence of an attack – "YES", the second output for the normal behavior – "NO"), or six outputs for attack classification (five outputs for the attacks, and the sixth output for the normal behavior). Three layers (input, hidden, and output). The hidden layer has 20 nodes, alpha = 0.7, beta = 0.8, "tansig" function is used in the input layer node, "purelin" in the hidden and output layer nodes, 50 epochs. The training set consisted of 4000 records. The attack and the normal behavior records were evenly distributed in the training set [69, 70].

C. Results

Five different attacks were used in the training set. Normal behavior records were considered as an attack, thus total of six attacks were used in this stage. 50% of the training set consisted of the concentrated attack, i.e. the attack that had to be differentiated from the others. Other 50% were evenly distributed between other attacks, i.e. 10% per attack. For example, normal behavior records needed to be defined. 50% of the training set for this assignment consisted of the records of normal behavior and other 50% contained records of Smurf, Neptune, Satan, IP Sweep, and Back attacks. All records were in random order. Table V demonstrates the results of this experiment. As shown in the table, the accuracy for differentiating the attacks is quite high for both Neural Networks. The lowest accuracy is 91% for Satan and the highest is 100% for Smurf, Neptune, and Back. These results let us make a conclusion that attacks can be differentiated, thus identified [69, 70].

TABLE V
FIFE ATTACK DATASET RESULTS

Attack Name	**RBF Accuracy**	**RBF False Positives**	**MLP Accuracy**	**MLP False Positives**
Smurf	100%	0	99.5%	0
Neptune	100%	0	100%	0
Satan	91%	7%	97.2%	2%
IP Sweep	99.5%	0	99.9%	0
Back	100%	0	100%	0
Normal	98.0%	1%	96.8%	2%

For the second stage of the experiments NN with six outputs were used. At this level there was an attempt to create an IDS that is capable of classifying the attacks. A dataset of five attacks and normal behavior records was used. The attacks were evenly distributed in the dataset. Table VI demonstrates the result of this experiment. As we can see the accuracy of classifying attacks is 93.2% using RBF Neural Network and 92.2% using MLP Neural Network. In most cases the Networks managed to identify an attack correctly. The false positive rate is very low in both cases, false negative rate is not high either, and the misidentified attacks rate (misclassification of the attacks) is 5%-6%. Overall, it is possible to conclude that both neural networks were capable of identifying the attacks [69, 70].

TABLE VI
RESULTS OF ATTACK CLASSIFICATION

	Accuracy	**False Positives**	**False Negatives**	**Misidentified Attacks**
RBF	93.2%	0.8%	0.6%	5.4%
MLP	92.2%	0	2.1%	5.7%

VIII. CONCLUSIONS

Reliable computer security to a large degree depends on development of biometric technology in general and HCI-based biometrics in particular. This affordable and non-intrusive way of verifying the user's identity holds a lot of potential to developing secure and user friendly systems and networks. As long as the issues of privacy are sufficiently addressed by the developers of HCI-based security systems commercial potential of development in this area is very substantial.

In this paper, we have demonstrated how expanding on techniques developed in the field of biometric recognition and combining results with methodology from intrusion detection systems can greatly increase computer security and prevent many popular types of attacks from going undetected. We have reviewed and analyzed a number of direct HCI-based biometrics as well as indirect HCI-based behavioral biometrics including audit logs, call-stack data, GUI interaction events, network traffic, registry access data, storage activity, and system calls. We have demonstrated experimentally how an intrusion detection system can be constructed based on analysis of observable human actions or outputs of computer applications, namely, network traffic dataflow.

In addition to computers, HCI-based biometrics are also well suited for verification of users which interact with cell phones, smart cars, or points of sale terminals. As the number of electronic appliances used in homes and offices increases so does the potential for utilization of this novel and promising technology. Also inclusion of additional input devices such as stylus, touch-pad, and digitizing tablet in the scope of HCI-based biometrics research will make the technology more applicable for the general public. Future research should be directed at increasing overall accuracy of such systems as well as looking into possibility of developing multimodal HCI-based biometrics as people often engage on multiple channels of interaction with a computer, for example using a mouse and keyboard simultaneously.

Acknowledgement

This paper is based upon work supported by National Science Foundation Grant No. DGE 0333417 ”Integrative Geographic Information Science Traineeship Program”, awarded to the University at Buffalo.

References

[1] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," IEEE Trans. Circuits Syst., Vol. 14., pp. 4-20, Video Technol, 2004.

[2] R. V. Yampolskiy, "Motor-Skill Based Biometrics," In Assuring Business processes, Proceedings of the 6th Annual Security Conference, Ed. G. Dhillon. Global Publishing, Las Vegas, NV, USA., April 11-12, 2007.

[3] R. V. Yampolskiy, "Human Computer Interaction Based Intrusion Detection," 4th International Conference on Information Technology: New Generations (ITNG 2007), Las Vegas, Nevada, USA, April 2-4, 2007.

[4] J. Ilonen, "Keystroke dynamics," Available at: www.it.lut.fi/kurssit/03-04/010970000/seminars /Ilonen.pdf, Retrieved July 12, 2006.

[5] A. K. Jain, R. Bolle, and S. Pankanti, "BIOMETRICS: Personal Identification in Networked Society," Kluwer Academic Publishers, 1999.

[6] F. Monrose and A. D. Rubin, "Keystroke Dynamics as a Biometric for Authentication," Future Generation Computing Systems (FGCS) Journal: Security on the Web (special issue), March 2000.

[7] F. Bergadano, D. Gunetti, and C. Picardi, "User authentication through keystroke dynamics," ACM Transactions on Information and System Security (TISSEC), November 2002.

[8] D. Gunetti, C. Picardi, and G. Ruffo, "Keystroke Analysis of Different Languages: a Case Study," Proc. of the Sixth Symposium on Intelligent Data Analysis (IDA 2005), Madrid, Spain, 2005.

[9] M. Curtin, C. C. Tappert, M. Villani, G. Ngo, J. Simone, H. S. Fort, and S. Cha, "Keystroke Biometric Recognition on Long-Text Input: A Feasibility Study," Proc. Int. Workshop Sci Comp/Comp Stat (IWSCCS 2006), Hong Kong, June 2006.

[10] G. Bartolacci, M. Curtin, M. Katzenberg, N. Nwana, S.-H. Cha, and C. C. Tappert, " Long-Text Keystroke Biometric Applications over the Internet," MLMTA, 2005.

[11] M. Pusara and C. E. Brodley, "User re-authentication via mouse movements," VizSEC/DMSEC '04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security, Washington DC, USA, 2004.

[12] A. A. E. Ahmed and I. Traore, "Detecting Computer Intrusions Using Behavioral Biometrics," Third Annual Conference on Privacy, Security and Trust, St. Andrews, New Brunswick, Canada, October, 2005.

[13] A. A. E. Ahmed and I. Traore, "Anomaly Intrusion Detection based on Biometrics," Workshop on Information Assurance, United States Military Academy, West Point, NY, June 2005.

[14] H. Gamboa and V.-. A. Fred., 2004., "A Behavioral Biometric System Based on Human Computer Interaction," In Proceedings of SPIE, 2004.

[15] H. Gamboa and A. Fred, "An identity authentication system based on human computer interaction behaviour," Proc. of the 3rd Intl. Workshop on Pattern Recognition in Information Systems, 2003.

[16] M. Orozco, Y. Asfaw, A. Adler, S. Shirmohammadi, and A. E. Saddik, "Automatic Identification of Participants in Haptic Systems," 2005 IEEE Instrumentation and Measurement Technology Conference, Ottawa, Canada, 17-19 May 2005.

[17] M. Orozco, Y. Asfaw, S. Shirmohammadi, A. Adler, and A. E. Saddik, "*Haptic-Based Biometrics: A Feasibility Study*," IEEE Virtual Reality Conference, Alexandria, Virginia, USA, March 25-29, 2006.

[18] M. O. Trujillo, I. Shakra, and A. E. Saddik, "Haptic: the new biometrics-embedded media to recognizing and quantifying human patterns," MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia, Hilton, Singapore, 2005.

[19] Caslon-Analytics, Available at: http://www.caslon.com.au/biometricsnote6.htm, Retrieved October 2, 2005.

[20] S. J. Stolfo, S. Hershkop, K. Wang, O. Nimeskern, and C.-W. Hu, "A Behavior-based Approach to Securing Email Systems," Mathematical Methods, Models and Architectures for Computer Networks Security, Sept. 2003.

[21] S. J. Stolfo, C.-W. Hu, W.-J. Li, S. Hershkop, K. Wang, and O. Nimeskern, "Combining Behavior Models to Secure Email Systems," CU Tech Report, Available at: www1.cs.columbia.edu/ids/publications /EMT-weijen.pdf, April 2003.

[22] E. H. Spafford and S. A. Weeber., "Software Forensics: Can We Track Code to its Authors?" 15th National Computer Security Conference, Oct 1992.

[23] A. Gray, P. Sallis, and S. MacDonell, "Software Forensics: Extending Authorship Analysis Techniques to Computer Programs," In Proc. 3rd Biannual Conf. Int. Assoc. of Forensic Linguists (IAFL'97), 1997.

[24] G. Frantzeskou, S. Gritzalis, and S. MacDonell, "Source Code Authorship Analysis for Supporting the Cybercrime Investigation Process," 1st International Conference on eBusiness and Telecommunication Networks - Security and Reliability in Information Systems and Networks Track, Setubal Portugal, August 2004.

[25] R. V. Yampolskiy, "Behavior Based Identification of Network Intruders," 19th Annual CSE Graduate

Conference (Grad-Conf2006), Buffalo, NY, February 24, 2006.

[26] R. V. Yampolskiy and V. Govindaraju, "Use of Behavioral Biometrics in Intrusion Detection and Online Gaming," Biometric Technology for Human Identification III. SPIE Defense and Security Symposium, Orlando, Florida, 17-22 April 2006.

[27] J. Ramon and N. Jacobs, "Opponent modeling by analysing play," Proceedings of the Computers and Games workshop on Agents in Computer Games, Edmonton, Albera, Canada, 2002.

[28] A. Brömme and S. Al-Zubi, "Multifactor Biometric Sketch Authentication," In A. Brömme and C. Busch, editors, Proceedings of the BIOSIG 2003, Darmstadt, Germany, 24. July 2003.

[29] S. Al-Zubi, A. Brömme, and K. Tönnies, "Using an Active Shape Structural Model for Biometric Sketch Recognition," In Proceedings of DAGM, Magdeburg, Germany, 10.-12. September 2003.

[30] D. Y. Yeung and Y. Ding, "Host-based intrusion detection using dynamic and static behavioral models," *Pattern Recognition*, vol. 36, pp. 229-243.

[31] J. Marin, D. Ragsdale, and J. Surdu, "A hybrid approach to the profile creation and intrusion detection," DARPA Information Survivability Conference and Exposition (DISCEX II'01), 2001.

[32] T. Lane and C. E. Brodley, "An Application of Machine Learning to Anomaly Detection," 20th Annual National Information Systems Security Conference, 1997.

[33] T. Lane and C. E. Brodley, "Detecting the Abnormal: Machine Learning in Computer Security," Department of Electrical and Computer Engineering, Purdue University Technical Report ECE-97-1, West Lafayette, January 1997.

[34] A. Garg, R. Rahalkar, S. Upadhyaya, and K. Kwiat, "Profiling Users in GUI Based Systems for Masquerade Detection," The 7th IEEE Information Assurance Workshop (IAWorkshop 2006), West Point, New York, USA, June 21-23, 2006.

[35] D. E. Denning, "An intrusion-detection model," IEEE Transactions on Software Engineering, Vol. 13(2), pp. 222-232, 1987.

[36] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: A rule-based intrusion detection approach," Software Engineering, Vol. 21(3), pp. 188-199, 1995.

[37] A. K. Ghosh, A. Schwartzbard, and M. Schatz, "Learning program behavior proles for intrusion detection," In Proceedings of the First USENIX Workshop on Intrusion Detection and Network Monitoring, April 1999.

[38] F. Apap, A. Honig, S. Hershkop, E. Eskin, and S. Stolfo, "Detecting malicious software by monitoring anomalous windows registry accesses," Technical report, CUCS Technical Report, 2001.

[39] A. G. Pennington, J. D. Strunk, J. L. Griffin, C. A. N. Soules, G. R. Goodson, and G. R. Ganger, "Storage-based intrusion detection: Watching storage activity for suspicious behavior," Technical report CMU--CS--02--179. Carnegie Mellon University, October 2002.

[40] H. H. Feng, O. M. Kolesnikov, P. Fogla, W. Lee, and W. Gong, "Anomaly detection using call stack information," In Proceedings of IEEE Symposium on Security and Privacy, 2003.

[41] G. Gupta, C. Mazumdar, and M. S. Rao, "Digital Forensic Analysis of E-mails: A trusted E-mail Protocol," *International Journal of Digital Evidance*, vol. 2, 2004.

[42] O. D. Vel, A. Anderson, M. Corney, and G. Mohay, "Mining Email Content for Author Identification Forensics," SIGMOD: Special Section on Data Mining for Intrusion Detection and Threat Analysis, 2001.

[43] A. R. Jansen, D. L. Dowe, and G. E., "Farr Inductive Inference of Chess Player Strategy," Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence (PRICAI'2000), 2000.

[44] C. Varenhorst, "Passdoodles; a Lightweight Authentication Method," Available at: http://people.csail.mit.edu/emax/papers/varenhorst.pdf , July 27, 2004.

[45] I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin, "The Design and Analysis of Graphical Passwords," Proceedings of the 8th USENIX Security Symposium, Washington, D.C., August 23-36, 1999.

[46] K. Renaud, "Quantifying the Quality of Web Authentication Mechanisms. A Usability Perspective," Journal of Web Engineering, Vol. 0, No. 0, Available at: http://www.dcs.gla.ac.uk/~karen/Papers/j.pdf, 2003.

[47] M. Schonlau, W. DuMouchel, W.-H. Ju, A. F. Karr, M. Theus, and Y. Vardi, "Computer Intrusion: Detecting Maquerades," *Statistical Science*, vol. 16, pp. 1-17, 2001.

[48] R. A. Maxion and T. N. Townsend, "Masquerade detection using truncated command lines," In International conference on dependable systems and networks(DNS-02), 2002.

[49] V. Dao and V. Vemuri, "Profiling Users in the UNIX OS Environment," International ICSC Conference on Intelligent Systems and Applications, University of Wollongong Australia, Dec. 11-15, 2000.

[50] T. Goldring, "User Profiling for Intrusion Detection in Windows NT," *Computing Science and Statistics*, vol. 35, 2003.

[51] K. Kaufman, G. Cervone, and R. S. Michalski, "An Application of Symbolic Learning to Intrusion Detection: Preliminary Results From the LUS Methodology," Reports of the Machine Learning and Inference Laboratory, MLI 03-2, George Mason University, Fairfax, VA, June, 2003.

[52] R. V. Yampolskiy, "Indirect Human Computer Interaction-Based Biometrics for Intrusion Detection Systems," The 41st Annual IEEE International Carnahan Conference on Security Technology (ICCST2007), Ottawa, Canada, October 9-11, 2007 (to appear).

[53] T. Lunt, "Detecting Intruders in Computer Systems," In Proceedings of the 1993 Conference on Auditing and Computer Technology, 1993.

[54] A. Wespi, M. Dacier, and H. Debar, "Intrusion Detection Using Variable-Length Audit Trail Patterns," In Recent Advances in Intrusion Detrection (RAID), 2000.

[55] J. P. Anderson, "Computer Security Threat Monitoring and Surveillance," Technical Report. James P. Anderson Company, Fort Washington, Pennsylvania, April 1980.

[56] N. Ye, "A markov chain model of temporal behavior for anomaly detection," In Proceedings of the 2000 IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop, 2000.

[57] A. Seleznyov and S. Puuronen, "Anomaly Intrusion Detection Systems: Handling Temporal Relations between Events," Web proceedings of the 2nd International Workshop on Recent Advances in Intrusion Detection (RAID'99), 1999.

[58] C. C. Michael, "Finding the vocabulary of program behavior data for anomaly detection," DARPA Information Survivability Conference and Exposition, 2003, 22-24 April 2003.

[59] W. Lee, S. J. Stolfo, and K. W. Mok, "A Data Mining Framework for Building Intrusion Detection Models," IEEE Symposium on Security and Privacy, Okland, CA, 1999.

[60] Y. Li, N. Wu, S. Jajodia, and X. S. Wang, "Enhancing Profiles for Anomaly Detection Using Time Granularities," Journal of Computer Security, Vol. 10(1), pp. 137-157, 2002.

[61] C. Ko, G. Fink, and K. Levitt, "Automated detection of vulnerabilities in privileged programs by execution monitoring," In Proceedings of the 10th Annual Computer Security Applications Conference, December 1994.

[62] C. C. Michael and A. Ghosh, "Using finite automata to mine execution data for intrusion detection: A preliminary report," In Proceedings of the Third International Workshop in Recent Advances in Intrusion Detection, Toulouse, France, October 2000.

[63] H. Feng, O. Kolesnikov, P. Fogla, W. Lee, and W. Gong, "Anomaly Detection using Call Stack Information," Proceedings of the IEEE Security and Privacy, Oakland, CA, USA, May 11-14, 2003.

[64] J. Giffin, S. Jha, and B. Miller, "Efficient context-sensitive intrusion detection," In 11th Annual Network and Distributed Systems Security Symposium (NDSS), San Diego, California, February 2004.

[65] Y. Kim, J.-Y. Jo, and K. Suh, "Baseline Profile Stability for Network Anomaly Detection," IEEE ITNG 2006, Internet and Wireless Network Security track, Las Vegas, NV, April 2006.

[66] L. d. S. Silva, A. F. d. Santos, J. D. d. Silva, and A. Montes, "A Neural Network Application for Attack Detection in Computer Networks," *Instituto Nacional de Pesquisas Espanciais*, 2004.

[67] Z. Zhang and C. Manikopoulos, "Investigation of neural network classification of computer network attacks," Proceedings of International Conference on Information Technology: Research and Education, 11-13 Aug. 2003.

[68] R. Sommer and V. Paxson, "Enhancing Byte-Level Network Intrusion Detection Signatures with Context," Proc. of 10th ACM Conference on Computer and Communications Security, 2003.

[69] D. Novikov, R. V. Yampolskiy, and L. Reznik, "Artificial Intelligence Approaches for Intrusion Detection," Long Island Systems Applications and Technology Conference (LISAT2006). Long Island, New York., May 5, 2006.

[70] D. Novikov, R. V. Yampolskiy, and L. Reznik, "Anomaly Detection Based Intrusion Detection," Third International Conference on Information Technology: New Generations (ITNG 2006), Las Vegas, Nevada, USA, April 10-12, 2006.

[71] D. Novikov., "Neural Networks to Intrusion Detection.," MS thesis, Rochester Institute of Technology. Rochester, NY, October 2005.

[72] D. Liu and F. Huebner, "Application Profiling of IP Traffic," 27th Annual IEEE Conference on Local Computer Networks, Nov. 6-8, 2002.

[73] K. Thompson, G. Miller, and R. Wilder, "Wide area Internet traffic patterns and characteristics," In IEEE Network, Vol. 11, pp. 10-23, November 1997.

[74] M. Banikazemi, D. Poff, and B. Abali, "Storage-based intrusion detection for storage area networks (SANs)," Proceedings. 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies, 2005., 11-14 April 2005.

[75] Y. Zhang and D. Wang, "Research on Object-Storage-Based Intrusion Detection," 12th International Conference on Parallel and Distributed Systems (ICPADS), 12-15 July 2006.

[76] J. L. Griffin, A. G. Pennington, J. S. Bucy, D. Choundappan, N. Muralidharan, and G. R. Ganger, "On the Feasibility of Intrusion Detection Inside Workstation Disks," Technical Report CMU-PDL-03-106, Carnegie Mellon University, 2003.

[77] L.-c. Lam, W. Li, and T.-c. Chiueh, "Accurate and Automated System Call Policy-Based Intrusion Prevention," in Proceedings of 2006 International Conference on Dependable Systems and Networks (DSN 2006), June 2006.

[78] S. A. Hofmeyr, S. Forrest, and A. Somayaji, "Intrusion detection using sequences of system calls," Journal of Computer Security, Vol. 6, pp. 151-180, 1998.

[79] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: alternative data models," Proceedings of the 1999 IEEE Symposium on Security and Privacy, Oakland, CA, USA, May 9, 1999.

[80] N. Nguyen, P. Reiher, and G. H. Kuenning, "Detecting insider threats by monitoring system call activity," IEEE Systems, Man and Cybernetics Society Information Assurance Workshop, 18-20 June 2003.

[81] C. Marceau, "Characterizing the Behavior of a Program Using Multiple-Length N-grams," Proceedings of the New Security Paradigms Workshop 2000, Cork, Ireland, Sept. 19-21, 2000.

[82] D. Wagner and D. Dean, "Intrusion detection via static analysis," In IEEE Symposium on Security and Privacy, 2001.

[83] A. K. Ghosh, A. Schwatzbard, and M. Shatz, "Learning Program Behavior Profiles for Intrusion Detection," in Proceedings 1 st USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, California, April 1999.

[84] S. Bhatkar, A. Chaturvedi, and R. Sekar, "Dataflow Anomaly Detection," IEEE Symposium on Security and Privacy, May 2006.

[85] I. Koychev and I. Schwab, "*Adaptation to Drifting User's Interests*," In Proceedings of ECML2000 Workshop: Machine Learning in New Information Age, Barcelona, Spain, 2000.

[86] A. Tsymbal, "The problem of concept drift: definitions and related work," Technical Report TCD-CS-2004-15, Computer Science Department, Trinity College, Dublin, Ireland, 2004.

[87] R. V. Yampolskiy and V. Govindaraju, "Dissimilarity Functions for Behavior-Based Biometrics," Biometric Technology for Human Identification IV. SPIE Defense and Security Symposium, Orlando, Florida, April 9-13, 2007.

[88] P. T. Stanton, W. Yurcik, and L. Brumbaugh, "FABS: file and block surveillance system for determining anomalous disk accesses," Proceedings from the Sixth Annual IEEE Information Assurance Workshop, 15-17 June 2005.

[89] Poker-edge.com, "Stats and Analysis," Available at: http://www.poker-edge.com/stats.php, Retrieved June 7, 2006.

[90] DARPA, "Intrusion Detection Evaluation," MIT Lincoln Laboratory, Available at: http://www.ll.mit.edu/ist/ideval, 1998.

[91] W. Lee, S. Stolfo, and K. Mok, "Mining in a Data-Flow Environment: Experience in Network Intrusion Detection," In Proceedings of the 5th ACM SIGKDD, San Diego, CA, 1999.

[92] R. V. Yampolskiy, "Feature Extraction Methods for Character Recognition," Master’s Thesis. Rochester Institute of Technology, Rochester, NY, May 10, 2004.

Roman V. Yampolskiy (Student Member, IEEE) was born in Riga, Latvia, on August 13, 1979. He received the B.S./M.S. (High Honors) combined degree in computer science from Rochester Institute of Technology, Rochester, NY, USA in 2004.

He was a Research Assistant for the Laboratory for Applied Computing, department of computer science, Rochester Institute of Technology. Since 2004, he has been with the Center for Unified Biometrics and Sensors, University at Buffalo. He is currently a PhD candidate in the department of computer science and engineering at the University at Buffalo, Buffalo, NY. He is a National Science Foundation Integrative Graduate Education and Research Traineeship Program Fellow. He has a number of journal and conference publications describing his research in artificial intelligence, neural networks, genetic algorithms, pattern recognition and behavioral profiling.

Mr. Yampolskiy is a member of IEEE Communications Society.

Venu Govindaraju is a Professor of Computer Science and Engineering at the University at Buffalo (SUNY Buffalo). He received his B-Tech (Honors) from the Indian Institute of Technology (IIT), Kharagpur, India in 1986, and his Ph.D. from UB in 1992.

He has co-authored more than 230 scientific papers. He has been the PI/Co-PI of projects funded by government and industry for over 50 million dollars in the last 15 years. He is the founding director of the Center for Unified Biometrics and Sensors (CUBS) and the associate director of the Center for Document Analysis and Recognition (CEDAR).He has served on the editorial boards of five premier journals in his area including the IEEE Transactions on Pattern Analysis and Machine Intelligence. He has served as the general chair of the IEEE AutoID 2005 and is the program co-chair of the First IEEE Conference on Biometrics: Theory, Applications and Systems (BTAS) in September 2007.

Dr. Govindaraju is a Fellow of the IEEE and the IAPR.