25

Applications of keystroke analysis for improved login security and continuous user authentication

S.M.Furnell, J.P.Morrissey, P.W.Sanders and C.T.Stockel
Network Research Group
University of Plymouth, Plymouth, UK (stevef@soc.plym.ac.uk)

Abstract

This paper examines the use of keystroke analysis as a means of improving authentication in modern information systems, based upon the biometric measurement of user typing characteristics. The discussion identifies that the concept may be implemented in two ways, providing the basis for both an enhanced authentication front-end as well as for continuous, transparent supervision throughout the session.

Two practical systems have been implemented, based upon static and dynamic verification techniques. The static verifier uses a neural network approach, whilst the dynamic verifier involves statistical analysis methods. The effectiveness of each module is examined using experimental test subject groups. The results observed allow the strategies to be contrasted, with a general assessment of the protection that the combination of techniques would afford.

The paper also discusses how the techniques could be integrated within a more comprehensive intrusion detection framework, capable of identifying various classes of abuse.

Keywords

Keystroke Analysis, Intrusion Detection, Authentication, Biometrics, Neural Networks.

1 INTRODUCTION

The issues of user identification and authentication are of paramount importance in the provision of secure information systems. If a user is not identified, it is impossible to grant specific access rights or ensure individual accountability for activities. Without authentication, systems could be used by unauthorised parties for illegitimate purposes. Whilst obtaining a (claimed) identity is extremely straightforward, subsequent verification is problematic and various approaches exist. The potential foundations for authentication have beencategorised by Wood (1977) based on either something the user *knows* (e.g. a password), *has* (e.g. a card) or *is* (e.g. a biometric). The first method is currently the most widely used, with passwords having the advantages of both conceptual simplicity and easy

S. K. Katsikas et al. (eds.), *Information Systems Security*

implementation. However, passwords often provide an unreliable basis for authentication (Jobusch and Oldehoeft 1989) and stronger methods are, therefore, necessary. Biometrics are advantageous as they may not be easily guessed, stolen or transferred to other people. However, the cost of the technology required to implement most biometric methods largely precludes its uptake in many cases. What is, therefore, required is a biometric that can be obtained without requiring additional hardware. Fortunately, such a characteristic can potentially be identified in the form of users keystroke rhythm.

2 AN OVERVIEW OF KEYSTROKE ANALYSIS CONCEPTS

The premise of the approach is that typing characteristics will be reasonably unique, revealing individual user "signatures". Such characteristics may include inter-keystroke times, keystroke duration times, typing error frequency and force of keystrokes. Of these, the inter-keystroke time was found to be the best discriminator in a preliminary study and was, therefore, adopted for further practical investigations (keystroke duration and typing error frequency measures were evaluated but were found to give unacceptably high errors, rejecting legitimate users and accepting impostors. Force of keystrokes cannot be measured without a specially modified keyboard - something we wished to avoid as it would negate any cost savings).

Users are likely to differ dramatically in terms of typing styles and abilities (depending upon factors such as familiarity with the keyboard, experience and formal tuition) and their characteristics may be assessed to create an appropriate *profile* for use in subsequent authentication (with significant departures causing impostor alerts). Legitimate users will be expected to be consistent with this profile, although certain circumstances (e.g. hand injury, fatigue and keyboard variations) may affect performance.

Keystroke analysis may be incorporated into an authentication system in two ways - referred to as *static* and *dynamic* verification strategies.

2.1 Static verification

In this scenario authentication is based upon entry of a static text string. Such analysis could be used at two points; during the initial user login and whenever a volatile command is entered. The entry of username and password at login are ideal points for static keystroke analysis. The details would be entered as usual, but the system would also analyse the way in which it was typed, providing a further level of authentication. Volatile commands are any that may delete, modify or copy stored data (such as MS-DOS '*deltree*' or '*xcopy*'). These could be used to vandalise information, inconveniencing legitimate users and potentially causing serious damage. There would be a practical limit to the number of commands that could be monitored in this way as, for each one to be analysed, a specific profile would need to be obtained from the legitimate user.

Static verification appears to be the most common approach, having been the basis for a number of previous studies (Bleha et al. 1990; Joyce and Gupta 1990).

2.2 Dynamic verification

Using this approach authentication is based upon any arbitrary text input, allowing greater scope for real-time supervision during user sessions. Monitoring could occur continuously, providing a transparent means of authentication that is not possible with most other methods (even those based on

biometrics). The fact that authentication no longer relies upon a single judgement should prevent impostors from using unattended logged-in terminals.

2.3 Measures of effectiveness

As with other biometric systems, the effectiveness of keystroke analysis may be judged on the basis of the False Acceptance Rate (FAR) and the False Rejection Rate (FRR). The FAR relates to errors where impostors are falsely believed to be legitimate users. Conversely, the FRR refers to errors where the system identifies the legitimate user as an impostor. These rates share a mutually exclusive relationship and, as such, it is not possible to attain optimum levels for both measures. An "equal error" scenario is not really an appropriate compromise and a decision must, therefore, be made as to which rate should receive priority - which will vary depending upon whether static or dynamic authentication is used. In the static scenario, minimisation of the FAR is considered to be the priority, as any successful impostor could potentially go unchecked for a whole session. However, in the dynamic scenario, a greater window for impostor detection is available and so minimising the FRR becomes the most important consideration (as rejections *during* a session could upset a legitimate user more significantly than occasional false login failures). A further important consideration in the dynamic scenario is the speed with which the identity assessment can be provided by the system.

3 PRACTICAL STUDIES OF KEYSTROKE ANALYSIS

The research team has conducted two practical studies to evaluate the keystroke analysis concept, with test subjects ranging from experienced typists to comparative novices. Both investigations utilised PC-based experimental systems and related to the static and dynamic verification approaches respectively.

The systems shared common core functionality, using PC hardware interrupts to detect keyboard actions and collect inter-keystroke timings with one millisecond accuracy (the PC Keyboard Action interrupt (int 09h) is activated for each key depression and release, whilst the Timer interrupt (int 08h) runs for every "tick" of the programmable system clock. The combination of these interrupts, with the clock set to 1000 ticks per second, allowed collection of inter-keystroke timings of the desired resolution). The systems differed in terms of the analysis strategies employed; the static analyser utilised neural network techniques, whilst the dynamic system used statistical methods.

In both cases, analysis was restricted to character pairs (or *digraphs*) involving alphabetic and "space" keystrokes, as these were considered the most likely to reveal any characteristic rhythm and were also found to produce the best results in a previous study which conducted a comprehensive investigation of this aspect (Leggett and Williams 1988). Both systems attempted to ensure representative typing profiles by ignoring erroneous inputs. The profiler for the static analyser achieved this by totally disregarding inaccurate samples, whereas the version for the dynamic system (where samples were longer and, hence, abandonment would have been impractical) filtered out and ignored any deleted keystrokes.

Specific details of the two systems and their results are described in the sections the follow.

4 STATIC KEYSTROKE ANALYSIS

The static analyser was implemented using a neural network for pattern recognition, based upon a multi-layer perceptron network. The perceptron is one of the best studied single layer neural networks (McClelland and Rumelhart 1986) and is made up of several simple biological neuron models. The simple model is constructed from input lines, a thresholding unit and an output line. If inputs to the thresholding unit exceed a threshold value, the perceptron "fires", as shown in figure 1.

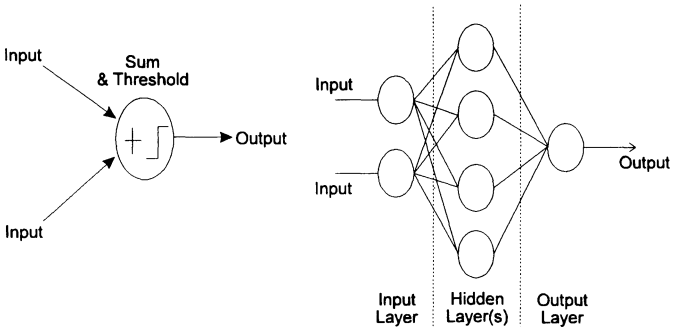

Figure 1 Single and multi-layer perceptrons.

However, the single layer perceptron is only able to classify linearly separable problems and the variability of user typing styles suggests that keystroke analysis does not fit into this category. As a result, a multi-layer perceptron was used, which is capable of solving nonlinear problems. These networks are made up of several neuron models, with at least one layer which is not directly connected to the input or output of the network. The version used in the experiment employed the back-propagation learning algorithm (McClelland and Rumelhart 1986), which attempts to minimise the error of the network by comparing the actual output to a target output for specific input vectors and adjusting the weights of input lines accordingly.

A *sampler* program collected samples from 15 experimental test subjects, with inter-keystroke times being stored for later use by the pattern recognition module. Each subject typed 35 samples of the following phrases :

1. REF 1 : *"PRINT"*;
2. REF 2 : *"TRANSFERENCE"*;
3. REF 3 : *"RED SKY AT NIGHT"*;
4. REF 4 : *"RUGBY PLAYERS ODD SHAPED BALLS"*.

These were chosen to allow sufficient overall representation of left-hand, right-hand and crossover digraphs. Differing phrase lengths were used to allow an examination of the effect that this had on attempts to classify the user's samples.

Set phrases were used to make the classification as difficult as possible, giving a worst case for the login scenario, where all users have the same password, but with a real life scenario for volatile commands. The use of individually distinct phrases such as usernames will be investigated in future

experiments but, because everyone is familiar with typing their own names, it is reasonable to assume that individual typing techniques will be more prominent and, therefore, more easily identifiable.

In the experiment every user had their own multi-layer perceptron for each phrase, to identify the specific user from the others sampled. An initial investigation found that the best topography for the neural network was that a single node be present within the input layer for each keystroke timing used, that the hidden layer have twice the nodes of the input layer and that a single node be used within the output layer. Twenty-five samples of the user to be identified were placed into the training set of the multi-layer perceptron, along with 25 from each of the other users to provide false samples. The remaining 10 samples from each user were used to evaluate the success that each network attained in identifying the legitimate user and rejecting impostors. This gave a total of 150 legitimate user attempts and 2100 impostor attempts.

Each neural network was presented with 4000 inter-keystroke samples individually for supervised training, chosen at random from the training set. The inter keystroke timings were presented to the input layer as measurements in seconds with no additional filtering or encoding (a future paper will discuss the application of genetic algorithms to data optimisation.) The resulting network was then evaluated, with the output being a value between 0 and 1 (with 1 indicating positive user identification). In actual fact, three decision boundaries (i.e. the points at which the output would be taken as a positive identification) were evaluated, namely 0.7, 0.8 and 0.9.

A low FRR was yielded if training was biased towards using more samples of the user to be identified. A ratio of four true samples to one false gave the results shown in table 1.

Table 1 Effectiveness with 4:1 training ratio

Phrase	*0.7*	*0.8*	*0.9*
REF 1	24	21	15
REF 2	8	6	5
REF 3	8	6	5
REF 4	4	3	2
Average	11	9	6

FAR given as %

Phrase	*0.7*	*0.8*	*0.9*
REF 1	10	15	26
REF 2	15	19	25
REF 3	21	23	27
REF 4	28	32	40
Average	19	22	30

FRR given as %

However, given that static analysis at login could be based upon the entry of two phrases (i.e. username and password), it was decided to evaluate effectiveness in this context. This was achieved using a combination of the REF 2 and REF 3 phrases, giving the results in table 2. Authentication confidence was classified at three levels :

- rejected, where a sample pair is rejected by both neural nets;
- low, where a sample pair is rejected by one neural net;
- high , where a sample pair is accepted by both neural nets.

Table 2 Results for two phrase combination

Threshold	*0.7*	*0.8*	*0.9*
Rejected	87	89	92
Low	13	11	8
High	1	0	0

Impostor attempts (%)

Threshold	*0.7*	*0.8*	*0.9*
Rejected	4	5	7
Low	38	10	50
High	58	55	43

User attempts (%)

It is important to reduce the FRR so that users are not overly inconvenienced through multiple authentication requests. However, in a static verification system it is more important for the FAR to be minimised, to prevent malicious attackers gaining initial entry to the system. If the reference phrases used are short, then a relatively high FRR of around 10% may be acceptable.

The details of this study are described in more depth in Morrissey (1995).

5 DYNAMIC KEYSTROKE ANALYSIS

The other study concerned a *dynamic* verification approach, with analysis being based upon statistical methods rather than neural networks. The profiles stored mean and standard deviation values for each sampled digraph.

A total of 30 test subjects were involved in this study, with each being required to submit a profile sample and two additional test samples. Profiling required users to enter two samples of a 2200 character *reference text*. A more significant length was necessary here to ensure that each users "natural" typing style emerged and that sufficient samples of each digraph were obtained to enable mean and standard deviation values to be established. The two further samples (of 574 and 389 characters) were used to represent impostor attempts by comparing them against all profiles not belonging to the same user. As such, the results in this phase were derived from more than 1700 impostor attempts.

The monitoring system compared inter-keystroke times from the test samples against user profiles, with incompatible times being judged invalid. These judgements were then analysed to detect impostors by (a) monitoring the percentage of invalid keystrokes during the 100 most recently typed and (b) monitoring the number of consecutive invalid keystrokes.

However, it was recognised that even legitimate users would generate some degree of invalid keystrokes and, as a consequence, each profile held associated *authentication thresholds* for these factors, with intrusion alerts being generated if either was exceeded during monitoring.

Given that the dynamic analyser would be used for continuous monitoring, the minimisation of false rejections was viewed as a priority. It was considered that if the FRR could be totally eliminated, then what would then be observed would effectively be a "worst-case" FAR. To this end, the authentication thresholds in the profiles were determined for each subject on an individual basis (by observing the peak values from the comparison of the two additional typing samples against their profile). This ensured that thresholds were set such that the legitimate user would always pass. The aims of the study were, therefore, to determine the FAR and the speed of successful impostor detection.

The experimental system exhibited an overall FAR of 15% (based upon 13% for the first sample and 18% for the second). This would be less significant when considered with the initial authentication provided by the static analyser and the combination of the two methods would almost certainly defeat most impostors.

In the cases where detection was achieved, the other important consideration was how many keystrokes the impostor was able to enter beforehand. The experimental findings here are presented in figure 2, showing the percentage of impostors detected within five distinct keystroke ranges (with cumulative values also indicated).

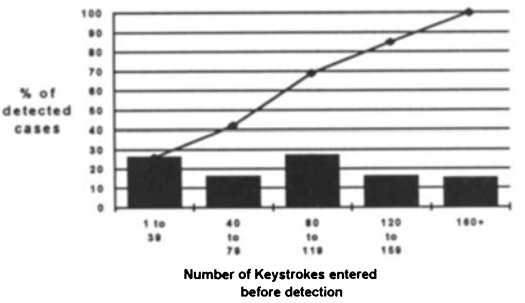

Figure 2 Keystrokes entered before impostor detection.

These results indicate that the vast majority of impostors would be detected within 160 keystrokes (the equivalent of two standard lines of text), with detection in under 40 keystrokes in 26% of cases. Whilst this may not combat the most destructive scenarios (e.g. the immediate entry of "delete *.*" might go unchallenged, unless specifically monitored as a *volatile command*), it should be sufficient to identify more common intruders, who generally require sustained access to effect a serious breach. Detection is likely to be quicker in cases where an impostor takes the place of a legitimate user, as a certain percentage of invalid keystrokes would already have been registered (by the legitimate user), causing the rejection threshold to be reached more easily.

A more detailed discussion of the dynamic analyser in terms of both the system implementation and the results observed is presented in Furnell (1995).

6 AN ASSESSMENT OF THE PRACTICAL STUDIES

An overall comparison of the experimental results is presented in table 3.

Table 3 Comparison of experimental study results

System	*Test subjects*	*FAR (%)*	*FRR (%)*
Static Analyser	15	8*	7*
Dynamic Analyser	30	15	0**

* based upon two phrase combination ** achieved via authentication thresholds

At this stage it has only been possible to evaluate the two systems independently, with the consequence that an overall FAR for the combined strategies could not be obtained. However, it is certain that in some cases where a false acceptance would occur within one analyser, the other could compensate by successfully trapping the impostor.

With regard to the performance of the individual systems, it can be observed that the neural network techniques used by the static verifier do not appear to yield such a significant improvement over the statistical approach as one might expect. However, it is expected that performance could be improved further by using a genetic algorithm to determine the specific digraphs that the network uses to identify

users. It is also recognised that as the analysers operate in different contexts, such a comparison is not exactly comparing like with like.

The single phrase results for the static analyser were rather disappointing in terms of the FRR observed. Whilst this would not pose a significant problem in the login context (and could be largely overcome by adopting two phrase analysis), it would be more problematic if monitoring volatile commands (which are normally single phrases). The results indicate that this could lead to repeated false rejections within a session, making supervision somewhat less than transparent. For this reason the technique is not suitable for this context unless it can be strengthened. In addition, the increased FRR with longer strings is worthy of further examination, although observation during the study revealed that test subjects generally made more mistakes with the longer samples, necessitating repeated re-entry, which potentially led to uncharacteristic typing.

With the dynamic analyser, the main point to recognise is that the FRR of 0% was obtained artificially. However, it is still envisaged that false rejections would not be frequent enough to significantly worry legitimate users in practice if authentication thresholds were set correctly.

Having established the effectiveness of the techniques, it is also interesting to consider how they could fit into a more comprehensive monitoring framework. This is discussed below, with the conceptual design of a system called IMS (Intrusion Monitoring System).

7 A WIDER INTRUSION DETECTION FRAMEWORK

Keystroke analysis alone is not a comprehensive basis for intrusion detection, as it only serves as an authentication mechanism. It is actually possible to identify several classes of system abuser (Anderson 1980) :

- external penetrators (i.e. unauthorised users of the system);
- masqueraders (i.e. authorised users who operate under the identity of another user);
- clandestine users (i.e. users who evade access controls and auditing);
- misfeasors (i.e. legitimate users who abuse their privileges).

Keystroke analysis would be ineffective against the latter two classes, given that these are legitimate users operating under their own identities. In addition, no defence would be provided against malicious processes (e.g. viruses). To provide more comprehensive supervision it is necessary to monitor other aspects of behaviour, along with general events that may suggest intrusions.

Behaviour profiling could be enhanced by also considering factors such as the time and location of system access, operating system command usage, application usage and data access. In addition, the system could monitor for events that might form part of a known intrusion scenario. The occurrence of these would be regarded as suspicious, especially in aggregation, and would be used to increase the *alert status* for the affected user or process. Some illustrative examples of such events are given below, along with the type of intruder that the occurrence of each would indicate :

- out of hours access (*penetrator* or *misfeasor*);
- use of dormant accounts (*external penetrator* or *masquerader*);
- excessive use of system "help" facilities (*external penetrator*);
- modification of executable file (*malicious process*).

The use of profiles and rules in this way is based upon a similar premise to that of the IDES intrusion monitor (Lunt 1990) and various other detection systems (Mukherjee et al. 1994).

At a high level, the proposed IMS architecture is based upon a centralised *Host* handling the monitoring and supervision of one or more *Clients* running on local workstations. The Clients will collect relevant data on user and process activity and respond to any suspected intrusions detected by the Host. All behaviour profiles and generic rules would be maintained securely at the Host, which also handles all of the analysis and the main bulk of other processing associated with the supervision. By contrast, the Client involves no local data storage and acts almost exclusively as an agent of the Host. At a lower level, the system would be comprised of a number of functional modules, as shown in figure 3 and outlined below.

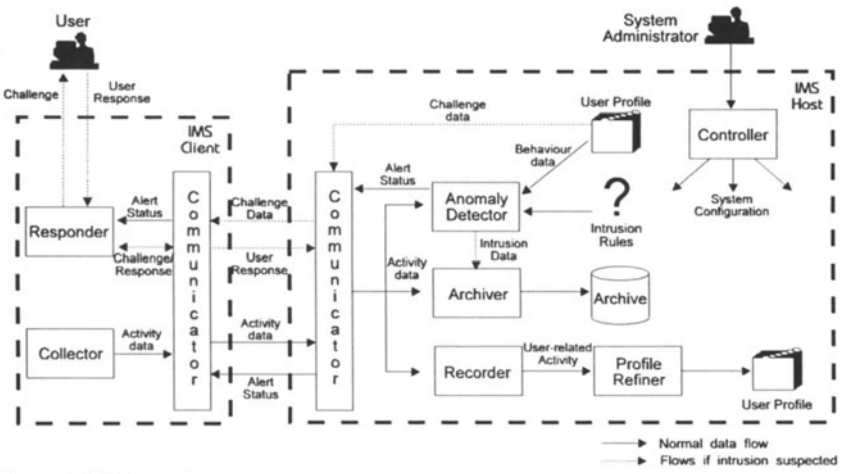

Figure 3 IMS Architecture.

The **Anomaly Detector** analyses user and process activity for signs of suspected intrusion, comparing it against both the behaviour profile for the current users (claimed) identity and the generic intrusion rules. It would be comprised of a number of further sub-modules, each handling a specific aspect of anomaly detection (e.g. keystroke analysis).

The **Profile Refiner** recognises the possibility that user behaviour may legitimately change over time and provides an automatic means for profiles to be updated (and, therefore, improved) to reflect this. The Refiner would be based upon the neural network approach, using the inherent ability to recognise patterns to identify behavioural characteristics that might not be apparent to human observers.

The **Recorder** handles the short-term storage of system activity data during the period of a user session. Upon termination the information will be picked up and used by the Profile Refiner, provided that the session was not considered anomalous.

The **Archiver** will collect data relating to all security relevant events and store it in a long-term archive (in the same manner as a traditional audit trail), providing a more permanent record of activities and suspected anomalies.

The **Collector** represents the interface between the IMS and underlying applications, with the responsibility for obtaining information on all relevant user and system activities (for comparison against behaviour profiles and generic rules). The resolution of data collection would be determined at the Host by the System Administrator.

The **Responder** resides in the Client and handles the task of responding to anomalies detected by the Host. The operation of the *Responder* would centre around the continuous monitoring of the alert status transmitted by the Host, with increases in the level triggering appropriate actions. The nature of response might include the issue of a user authentication challenge, suspension of a session or cancellation of a process.

The **Communicator** provides the interface between the Host and the local Client systems and, as such, its functionality is duplicated on both sides of the link. The principal functions would include transmitting activity information to the Host and then subsequently keeping the Client(s) informed of the current alert status. In a heterogeneous environment, the Client side would be responsible for resolving differences within the monitoring domain so that information could be presented to the Host in a consistent, standardised format.

The **Controller** allows the System Administrator to configure IMS operation. On the Host, this would apply to the *Anomaly Detector* (e.g. behaviour characteristics to consider / prioritise, generic rules in operation), the *Profile Refiner* (e.g. frequency of refinement) and the *Archiver* (e.g. level of detail required). On the Client, configuration would affect the operation of the *Collector* (e.g. the level of data collection) and the *Responder* (e.g. the response to each alert level). These settings would be controlled through the Host, with any Client settings being established during session initiation.

These modules would combine to provide an architecture offering improved security, whilst retaining the advantages of end-user convenience and financial viability.

8 CONCLUSIONS

The overall results from the experimental studies are encouraging and, although both systems exhibited some degree of error, it must be remembered that they are intended to supplement or strengthen security rather than provide a total solution. Both techniques have considerable potential for integration into existing systems, with the static analyser enhancing password mechanisms and the dynamic system enabling continuous supervision at subsequent times. This has already been achieved in a demonstrator system developed since the experimental study, which allows transparent monitoring of user activity (in real-time) on a client workstation.

Planned future development of the systems include the enhancement of the static analyser to incorporate a genetic algorithm, along with subsequent modification of the dynamic system to utilise these techniques if the results are suitably encouraging. The incorporation of keystroke analysis into an overall IMS-type framework is viewed as advantageous to provide more comprehensive supervision. Whilst the concept is not envisaged as a replacement for conventional authentication and access control methods, it will provide a way to strengthen existing systems and complement any security already provided.

9 REFERENCES

Anderson, J.P. (1980) *Computer Security Threat Monitoring and Surveillance*. James P. Anderson Co., Fort Washington, PA.

Bleha S.; Slivinsky, C.; and Hussien, B. (1990) Computer-Access Security Systems Using Keystroke Dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**, no.12, 1217-1222.

Furnell, S.M. (1995) *Data Security in European Healthcare Information Systems*. PhD Thesis. University of Plymouth, UK.

Jobusch, D.L. and Oldehoeft, A.E. (1989) A Survey of Password Mechanisms : Part 1. *Computers & Security*, **8**, no. 7, 587-604.

Joyce, R and Gupta, G. (1990) Identity Authentication Based on Keystroke Latencies. *Communications of the ACM*, **33**, no.2, 168-176.

Legget, J. and Williams, G. (1988) Verifying identity via keystroke characteristics. *International Journal of Man-Machine Studies* 28: 67-76.

Lunt, T.F. (1990) IDES: An Intelligent System for Detecting Intruders. *Proceedings of the Symposium : Computer Security, Threat and Countermeasures*, Rome, Italy.

McClelland, J.L. and Rumelhart, D.E. (1986) *Parallel Distributed Processing, Volume 1*. MIT Bradford Press.

Morrissey, J.P. (1995) *The Extension and Hardware Implementation of the Comprehensive Integrated Security System Concept*. PhD Thesis. University of Plymouth, UK.

Mukherjee, B.; Heberlein, L.T.; Levitt, K.N. (1994) Network Intrusion Detection. *IEEE Networks*, **8**, no.3, 26-41.

Wood, H.M. (1977) *The use of passwords for controlled access to computer resources*. National Bureau of Standards Special Publication 500-9, U.S Dept. of Commerce / NBS.

10 BIOGRAPHIES

Dr Steven Furnell is a Research Fellow within the Network Research Group at the University of Plymouth. He graduated from the University with a first class honours degree in Computing & Informatics and a PhD in computer security. In addition to addressing intrusion monitoring issues, he has also been involved in the development of security guidelines for European healthcare establishments. His current research interests also include Integrated Service Engineering and mobile telecommunications.

Joseph Morrissey has just completed research for a PhD at the University of Plymouth. He graduated from the University of Sussex with a degree in Electronic Engineering and then moved into computer security within the Network Research Group. Through research for his PhD he has investigated security within distributed environments, real-time authentication techniques and cryptographic hardware.

Peter Sanders is the Director of the Network Research Group at the University of Plymouth. He is currently involved with research into security systems, network design and Integrated Service Engineering in association with industry and the European Union, where he is currently engaged on

ACTS and TAP projects. He is the author of a number of technical papers, reports and books in these areas.

Dr Colin Stockel is Reader in Computer Science at the University of Plymouth. He graduated from the University of London with a BSc in Physics and a PhD in High Energy Particle Physics. Since 1967 he has taught and carried out research in Computer Science, becoming a Fellow of the British Computer Society and a Chartered Engineer. His current research interests lie in the area of computer security and networking, simulation and mathematical modelling, in which he is the author of numerous publications.