Enhancing Operating System Authentication Techniques

P.S. Dowland and S.M. Furnell

Network Research Group, University of Plymouth, Plymouth, United Kingdom
e-mail: pdowland@plymouth.ac.uk

Abstract

The need for enhanced user authentication has been evident for some time; but has not been addressed at the operating system level to any degree. Whilst all mainstream operating systems offer some level of user identification and authentication, this is generally based on the username/password combination. Although a number of extensions to operating system security have been proposed (with some reaching implementation) none, as yet, have been integrated into the core operating system kernel. Although there are examples that extend the operating system security model with additional measures (e.g. plug-in fingerprint scanners), these merely extend the operating system security rather than replace it with a more secure version.

This paper will consider the need to improve operating system security focussing upon the enhancement of user identification and authentication. In particular, the security weaknesses of the Microsoft Windows NT environment will be considered, leading to a discussion of supervision techniques that may be integrated within the NT security model. Finally, the conceptual integration of an Intrusion Monitoring System (IMS) architecture is considered.

Keywords

User authentication, user supervision, security, intrusion monitoring, Windows NT.

Introduction

The most commonly used form of operating system user authentication is the username/password pair. In most systems, the allocation of passwords (and sometimes usernames) is entirely at the discretion of the users and, as such, is the cause of many security loopholes. The weaknesses of passwords as the primary form of user authentication have been documented in previous works (Jobusch and Oldehoeft, 1989; Cherry et al, 1992) and will not be covered in detail here. However, typical weaknesses include passwords being easily guessed, shared among users, the use of dictionary words (which are more vulnerable to attack) and being written down near PCs. Even when passwords are more selectively chosen, they are still vulnerable to brute force attack, especially with the fast processors and distributed password cracking software now freely available (Savill, 1999).

It is clear that the 'out of the box' configuration for an operating system is inadequate for most systems. For example, most UNIX installations leave many security 'back-doors' into the system wide open by default (e.g. default password settings that administrators *should* change, but often do not), which provide an easy target for hackers (Stoll, 1989). Similarly, a standard installation of Microsoft Windows NT requires many steps before it can be considered secure (Microsoft, 1999a). Relying on passwords in their common form is inadequate and, therefore, some form of advanced user identification is desirable. Ideally, this should also be combined with some form of user monitoring; thus ensuring that a user's session cannot be hijacked. Hijacking occurs where a users' active session is taken over by another user (intruder). This can occur on a number of levels; firstly an intruder can simply resume a session by waiting for the user to leave their desk and then taking advantage of an

unprotected computer. Alternatively, an intruder may connect a device (computer) to the target computers' network connection and masquerades as the target computer. Whilst hijacked sessions are most likely to occur in a corporate networked environment, there are still risks to SME's and individuals – this is especially true with the trend towards e-commerce and the increased confidence in purchasing on-line (NOP, 1999). An intruder may be able to capture a credit-card purchase and then either modify or replay that same exchange of data to their advantage. Enhancing user authentication is, therefore, of value to both the commercial and private sectors.

Another problem, which is often overlooked during the selection of appropriate security systems, is that of internal misuse of computer systems. Most systems rely on the username/password pair to identify and authenticate a user. Once this authentication has been given, the user is often free to access the system without further checks or monitoring. Whilst most systems offer the ability to selectively exclude users and/or groups from specific shared resources, this is not usually the default setting. For example, under Windows NT, shares are, by default, accessible to all users and an administrator must specifically set access rights to ensure a shared resource is protected from internal misuse. A similar issue relates to private use of computing resources. Although this is not usually considered to be a security risk, it can represent a loss to a business either through physical resource usage or loss of computer processing time. Often the biggest loss to a company is that of lost employee time; not just through the time lost by the employee concerned but also in the time taken to investigate the problem and prevent further misuse (Audit Commission, 1998).

Operating system security weaknesses

With operating systems such as Microsoft's Windows NT4 comprising several million lines of code, it is, perhaps, no surprise that security weaknesses should occur. However, it is often surprising to see the scope and frequency with which such fundamental flaws are found. Using Microsoft Windows NT4 as an example, the Microsoft Product Security Notification Service issues several warnings each week, each identifying a potential security problem with the operating system or its sub-components (Microsoft, 1999b). Of course, Microsoft Windows NT is not the only operating system to suffer with such security problems – the many flavours of Unix also generate hundreds of security patches each year (see http://www.faqs.org/faqs/computer-security/most-common-qs/index.html). However, the wider distribution of Windows means that the consequences of security vulnerabilities are potentially more wide reaching. A further drawback with a "popular" operating system is that as its popularity increases, it becomes a greater target to hackers partly due to the increased usage (and, therefore, potential targets) but also because of the greater availability of information relating to security weaknesses. This has been particularly prevalent with the appearance of "script-kiddies" (young inexperienced hackers), who frequently use the many resources (called "filez") which are available from hacking sites on the Internet. A noticeable side effect of this is the use of alternative operating systems where security is of prime concern. For example, the US Army has switched to a MacOS-based web server platform, following a hacking incident when the server was running Windows NT (Donoghue, 1999). This is not to say that MacOS is any more secure than Windows NT, just less widely targeted.

Despite the frequency of these vulnerabilities, the only standard form of security provided by these operating systems for authentication purposes is the password.

Enhancing Windows NT security

Windows NT security can be considered on two levels, local machine and domain or remote login (Figure 1).

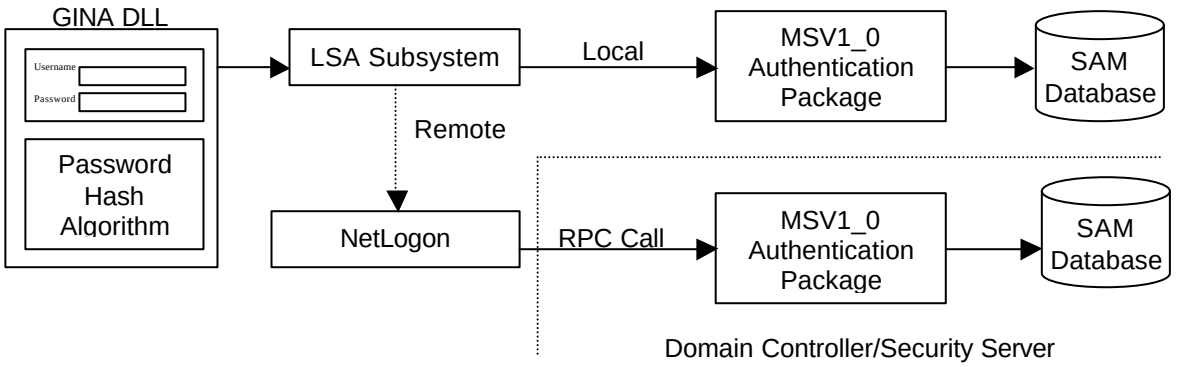

Figure 1 Local/remote user authentication

When a local user presses the "Control-Alt-Del" combination to initiate a login they are prompted to enter their username/password pair. The NT hash algorithm is then applied to the password and is passed on to the Local Security Authority (LSA) which calls the MSV1_0 authentication package. This hash is finally compared with the hash stored in the local Security Account Manager (SAM) database by the authentication package. Once a users' password is authenticated, an access token is issued that is valid for that users' session.

When a user wishes to be authenticated across a network (to log-in to a domain controller or for access to a remote machine), the password hash must be transferred across the network. When the user is prompted for their username/password they are also required to enter a valid domain. When the authentication package identifies that the account is not held locally, a call is made to the NetLogon service which sets-up a secure Remote Procedure Call (RPC) session to the domain controller to authenticate the login. The domain controller then issues a 16-bit challenge (the nonce). This challenge is then encrypted together with the password hash and is returned to the domain controller for authentication. Finally, the domain controller returns an access token which is valid for that users' session.

One of the main problems of the above technique is that once the challenge (nonce) has been intercepted and with knowledge of the encryption algorithm it is possible to determine the password hash. Given a known hash, it is feasible (with today's technology) to guess (using a dictionary and/or brute-force attack) the original password.

To achieve a more comprehensive approach under Windows NT would require a replacement GINA **G**raphical **I**dentification a**N**d **A**uthentication DLL (core user login system library e.g. username/password prompt). The GINA DLL provides an interface through which a user can provide his/her identification. This typically takes the form of the traditional username/password, but can be replaced with any form of identification (e.g. fingerprint scanner, iris scanner etc.).

There are a number of "add-on" software/hardware packages that can be used to enhance Windows NT security. One of the most common packages currently available is the fingerprint scanner. This is a small device that connects to the PC and provides a cost-effective way of authenticating a login attempt. These devices typically provide an additional

security module that integrates into the NT security model. Similar devices are also available to capture handprint geometry, facial patterns and there are devices appearing that are capable of iris scanning. Although these packages allow the enhancement of NT security by removing the need for the user to remember a password, they are not completely integrated into the operating system and only provide a replacement for the username/password prompt. There is also a significant cost overhead to be considered (for example, a fingerprint based authentication system would require the purchase of sufficient scanners for all the PC's in an organisation). Many of these solutions also depend on additional hardware that is dedicated to the task of providing enhanced authentication and, therefore, provides no additional benefit to the organisation concerned (i.e. no purpose other than security).

Even if these techniques were integrated into the NT security model, there are still gaps which leave significant security weaknesses. For example, even with a fingerprint scanner, once the user has logged-in using their finger, there is no guarantee that the same user will sit down and continue with the session. Similarly, if a user leaves their workstation, there is no means of checking if the user who continues the session is the same that started it. (Although all versions of Windows allow the configuration of a screensaver with password protection, this is not set by default. It should also be noted that the computer is unprotected from the time the user leaves their desk to the point at which the screensaver is activated, unless they explicitly lock the terminal). Due to these risks, some form of ongoing user supervision is required to ensure that the current user is the same as the user who activated the session. The remainder of this paper considers the adoption of an Intrusion Monitoring System (IMS) and the technical aspects involved in integrating into the Windows NT security model.

Description of an IMS

Following previous research work, a proposed IMS architecture is shown in figure 2. The specific functionality of this architecture has been described in a previous paper and will not be described in detail here (Furnell et al, 1997). At the basic level, the approach involves an IMS host monitoring activity occurring on a series of client systems. The client/server relationship of the IMS architecture shown fits neatly into the Windows NT security model architecture and the proposed IMS integration is described later in this paper. Further research work is necessary to fully integrate the IMS architecture into the Windows NT security model and will be the subject of a later paper.

The **Anomaly Detector** analyses the data gathered by the IMS client for signs of suspected intrusion. This data can be compared against both the user's behaviour profile and the generic intrusion rules (i.e. attack signatures).

The **Profile Refiner** allows the automatic modification of a user's profile in response to a valid session profile. This recognises the fact that a user's behaviour pattern may change over time (e.g. in a scenario where typing style has been profiled, their typing skill may improve) and allows a user's profile to evolve. Due to the nature of the data and the difficulty in recognising gradual behavioural pattern changes, it is likely that this would be implemented using some form of neural network (Furnell, 1994).

The **Recorder** stores a temporary record of system and user activity during a session (session profile) which can be used by the Profile Refiner to update the user profile, providing the session was not considered anomalous.

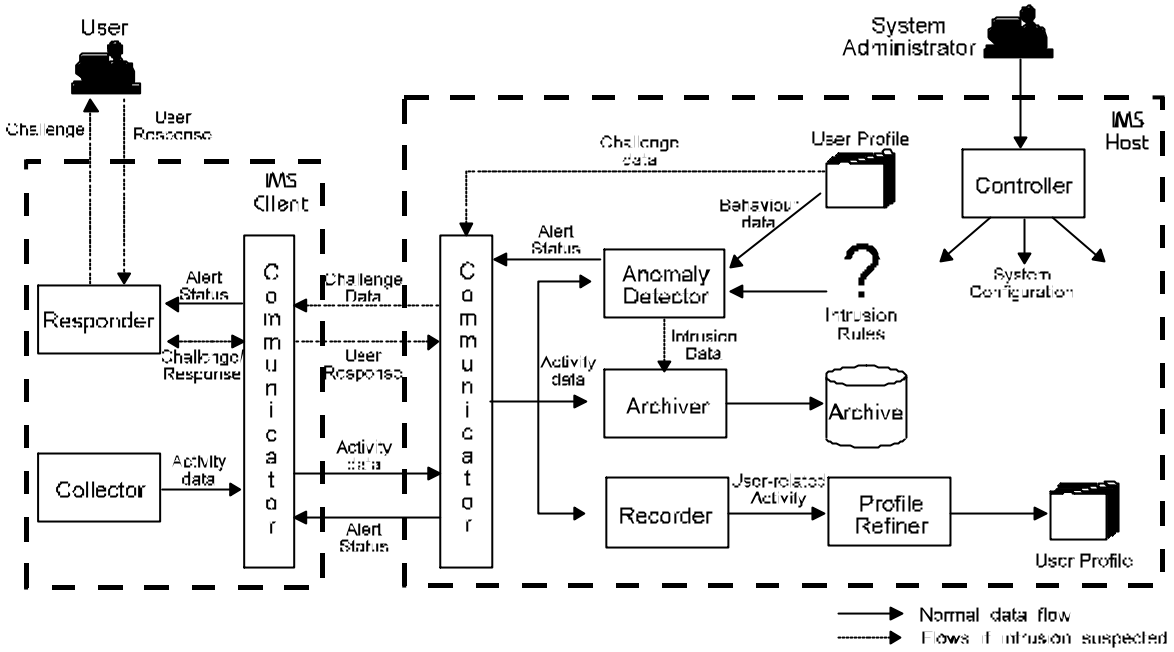

Figure 2 Proposed IMS Architecture

The **Archiver** provides an audit log, storing all security relevant events. This could also be extended to monitor *all* events if an organisation requires a more detailed log of user activity (e.g. to monitor user performance).

The **Collector** provides an interface between the IMS client and the applications running on the client computer. The collector is responsible for gathering information relevant to the user and his/her system activities. Under Windows NT the collector would be implemented as a mediator, collecting information gathered by low-level system functions that intercept system messages (e.g. keystrokes, mouse movements etc.) and forwarding this information on to the communicator.

The **Responder** provides user interface between the IMS software suite and the end-user. Its main task is that of monitoring the signals send from the server to the client and taking appropriate action where necessary. Possible actions include; issuing a user authentication challenge, suspending a session, limiting a user's actions or cancelling a process.

The **Communicator** provides the interface between the client and server IMS software. The communicator is responsible for ensuring a consistent, reliable and secure exchange of data between the client and server. Where an IMS system is implemented in a heterogeneous environment, the communicator is also responsible for data translation to provide consistent data formatting between different client platforms.

The **Controller** provides a management interface to the IMS server software allowing an administrator to configure the IMS system-operating parameters. The controller also allows an administrator to configure client-monitoring characteristics on a global, group, machine or individual user basis.

An Intrusion Monitoring System incorporates identification and authentication of users, monitoring of users for unusual behaviour or characteristics, together with the ability to modify the profile of a user to reflect changing patterns of use/behaviour. An IMS can rely on many physiological characteristics of the user (e.g fingerprint, voice etc.) and can also monitor behavioural traits such as keystroke patterns, mouse dynamics and

application/resource usage. However, it should be noted that the majority of commercially available IMS systems rely on traditional methods of user authentication

A strong potential candidate for a monitoring characteristic is that of keystroke analysis. This is a particularly attractive characteristic, as it requires no additional hardware (cost) or proprietary drivers (development time). By monitoring a user's typing profile it is possible to determine, with some accuracy, the identity of the current user. The use of a users' typing pattern as an authentication characteristic has been described in a number of papers (Furnell et al, 1996; Brown and Rogers, 1993) and has shown to be a strong distinguishing factor in certain contexts with overall False Acceptance Rate (FAR) figures as low as 4.2% being observed.

Although keystroke analysis is a good characteristic upon which to base user authentication, there are limitations. One of the major drawbacks of this characteristic is the very fact that users have a broad range of typing patterns. An inexperienced typist will use a keyboard in a slow deliberate manner, having a slow typing rate and most probably a high error rate. A trained touch-typist will type quickly with a low error rate. However, most inexperienced typists will type equally slowly and most touch-typists will type equally quickly. It is quite possible that the inter-keystroke time will be such that two typists may be indistinguishable in normal working environments.

Keystroke analysis may also be inappropriate depending on the environment in which it is used. For example, if a user is typing in numeric data for a prolonged period, it may be impossible to achieve a statistically valid sample of keystroke data upon which to base the authentication judgement. Similarly, if a user were drawing with a mouse, there would be no keystrokes to analyse.

From this, we can see that a composite approach is needed, where several appropriate authentication and monitoring techniques are applied. For example, a user may be initially authenticated by their fingerprint, after which their typing profile and application usage can be monitored. Similarly, if that user then starts to draw using the mouse, data can be recorded to determine if the dynamic movement of the mouse is consistent with the users' profile. This technique can also be applied where users *hotdesk*. If a user moves to a desk with an additional security-relevant device (e.g. a camera for faceprint recognition), the additional measures can be detected during an audit and then utilised for that user depending upon the settings in their profile.

Integrating an IMS into the Windows NT security model

If we consider the concept of an IMS, the username/password pair could be used to identify the user with a partial degree of certainty, whilst the continuously evaluated characteristics would allow the user to be monitored throughout the session. Using the previous example of keystroke analysis, a users' typing pattern can be monitored throughout the active session and compared with a historical profile. Deviation from this profile can be flagged and a threshold set beyond which further authentication of the user would be required (Furnell, 1995). This trust level can also determine the frequency of monitoring and, where further authentication is considered necessary, the degree of certainty needed (and, hence, the form of authentication to request).

To achieve an Intrusion Monitoring System (IMS) under Windows NT would require a replacement GINA DLL and an additional piece of software to provide the required

continuous monitoring together with a remote security server. A security server (or some form of centralised system) would be used to store, maintain and update the user profiles. This server would (in an ideal system) process all authentication requests together with local system audits and updates to profiles. This role is slightly different to that of a network server, which, usually, only authenticates requests for access at the beginning of a session. Instead, the security server would be responsible for ongoing authentication of a user throughout a session.

A user login would be performed locally (or remotely via a domain controller) and once the user's credentials are confirmed the monitoring program would be loaded to provide continuous user authentication (Figure 3). To prevent tampering, the IMS system would store user profiles remotely on a security server. The profiles would be encrypted and downloaded at login to the local computer (although for higher security the profiles could be maintained on the server, with authentication requests being handled by the server). To also offer security for the hardware (to ensure monitoring hardware had not been removed) a local machine audit can also be initiated, together with checks for dependent entries in configuration files or registry keys. An IMS system would also allow updating of the user profiles, to take into consideration changing user behaviour (e.g. keystroke patterns, application usage etc.) or appearance (e.g. facial recognition).

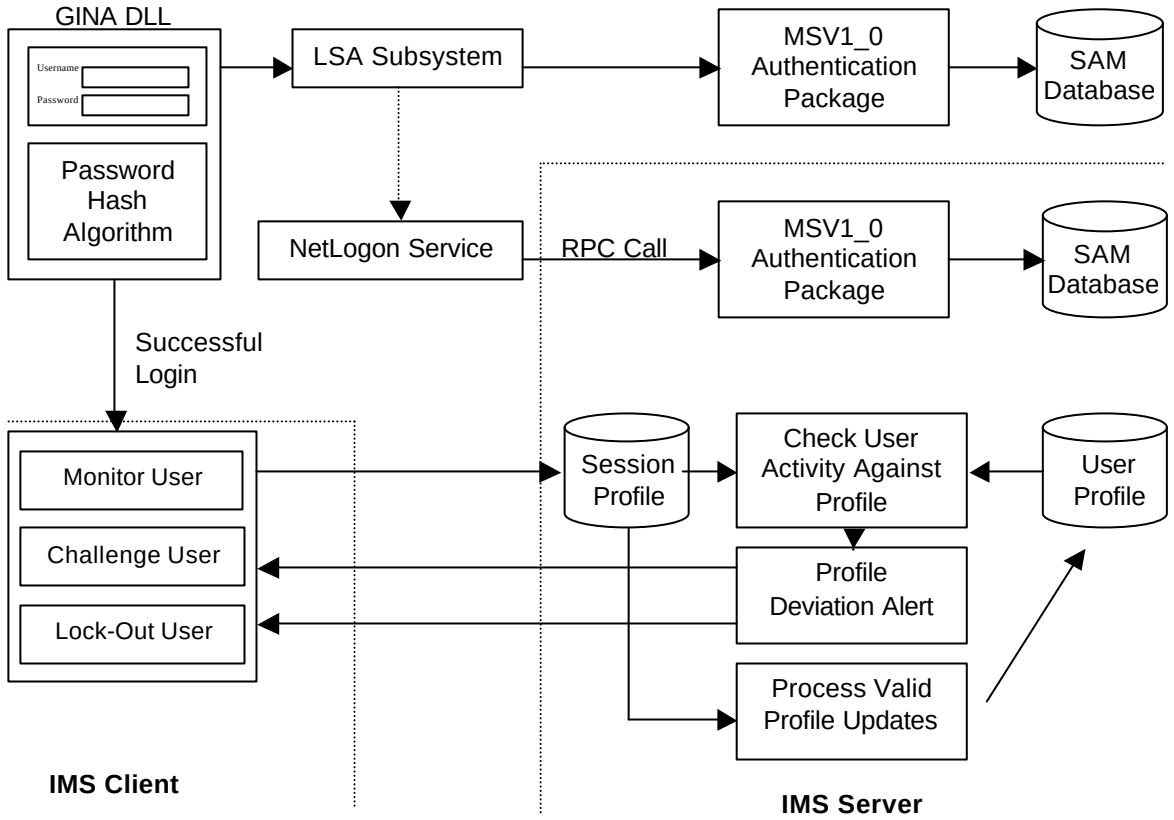

Figure 3 Prototype IMS-NT Integration

To reduce network traffic, it is envisaged that the user authentication would be performed on the local computer with only warnings or profile updates being fed back to the security server. Under certain scenarios it may be necessary to lock local computers if contact is lost with the security server to ensure an intruder had not removed a computer. However, it should be noted that this creates a weak point and appropriate measures will be needed to prevent a single server stopping the entire network, this could take the form of a backup

server (in a similar fashion to a secondary DNS server in an Internet context). Alternatively, the range of facilities available to the user can be restricted until the user can be re-authenticated. Another possible weak-point is the profile update process. It is important that the profile update is only performed once a user authentication confidence level is exceeded and it is established that the computer concerned has not been tampered with. In the event that a users' authentication threshold has been uncertain and/or the computer may have been tampered with, any proposed changes to the user profile should be discarded.

One of the most important factors in the implementation of continuous user monitoring is ensuring the transparency of the monitoring process. A system that requires users to continuously re-authenticate themselves will not be successful. Therefore, an IMS should allow background monitoring of an authenticated user, only interrupting the user in the event that further authentication is necessary (e.g. in the form of a challenge-response question).

Clearly an IMS system can provide enhanced user authentication. However, there is no single system configuration that will meet all the needs of all the users. Instead configuration of the security server and client monitoring software is dependent on the level of security required by the organisation and amount of inconvenience that is tolerable to the users (the classic False Acceptance Rate versus False Rejection Rate dichotomy) (Cope, 1990).

Conclusions

As the need for enhanced user authentication grows, operating systems will be extended to provide the necessary services. Windows NT already allows the use of a replacement GINA DLL, which allows OEM security vendors to supplement the Windows NT username/password login with additional/replacement authentication techniques. Alternative login techniques (e.g. fingerprint identification) allow the system confidence in user validity to be increased, but further security is needed to ensure the continued confidence in the user once past the initial login process. A process of continuous user authentication and monitoring, as described in the paper, is therefore desirable.

References

Audit Commission (1998), *Ghost in the Machine – An Analysis of IT Fraud and Abuse*, Audit Commission Publications, UK, ISBN 1-86240-056-3.

Brown, M. and Rogers, S. J. (1993), "User identification via keystroke characteristics of typed names using neural networks", *International Journal of Man-Machine Studies*, p999-1014.

Cherry, A., Henderson, M.W., Nickless, W.K., Olson, R. and Rackow, G. (1992), "Pass or fail: a new test for password legitimacy", Argonne National Laboratory, Mathematics and Computer Science Division, Paper Ref.: MCS-P328-1092,
http://www-proto.mcs.anl.gov/division/publications/abstracts/abstracts92.htm

Cope, J.B. (1990), "Biometric systems of access control", *Electrotechnology*, p71-74, April/May 1990.

Donoghue, A. (1999), "US Army scraps NT for MacOS", *Computing*, p14, 7th October 1999.

Fausett, L. (1994), *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*, Prentice-Hall International, New Jersey, USA, ISBN 0-13-042250-9.

Furnell, S.M. (1995), *Data security in European healthcare information systems*, PhD Thesis, University of Plymouth, UK.

Furnell, S.M., Morrissey, J.P., Sanders, P.W. and Stockel, C.T. (1996), "Applications of keystroke analysis for improved login security and continuous user authentication", *Proceedings of IFIP Sec '96*, Island of Samos, Greece, 21-24 May 1996, pp283-294.

Furnell, S.M., Illingworth, H.M., Katsikas, S.K., Reynolds, P.L. and Sanders, P.W. (1997), "A comprehensive authentication and supervision architecture for networked multimedia systems", *Proceedings of IFIP CMS '97*, Athens, Greece, 22-23 September 1997, pp227-238.

Jobusch, D.L. and Oldehoeft, A.E. (1989), "A survey of password mechanisms: Weaknesses and potential improvements. Part 1", *Computers & Security*, p587-603.

Microsoft Corporation Web Site (1999a), http://www.microsoft.com/security/issues/deployingc2.asp

Microsoft Corporation Web Site (1999b), http://www.microsoft.com/security/services/bulletin.asp

NOP Research Group (1999), "E-Commerce in Britain to reach £9.5 billion by 2000", http://www.nopres.co.uk/survey/internet/internet_item8.htm

Savill, J. (1999), *NT FAQ Web Site*, http://www.ntfaq.com/ntfaq/security21.html#security21

Stoll, C. (1989), *The Cuckoo's Egg*, Doubleday, New York.