A conceptual architecture for real-time intrusion monitoring

ABSTRACT

The detection and prevention of authorised activities, by both external parties and internal personnel, is an important issue within IT systems. Traditional methods of user authentication and access control do not provide comprehensive protection and offer opportunities for compromise by various classes of abuser. A potential solution is provided in the form of intrusion detection systems, which are able to provide proactive monitoring of system activity and apply automatic responses in the event of suspected problems.

The paper presents the principles of intrusion monitoring and then proceeds to describe the conceptual architecture of the Intrusion Monitoring System (IMS), an approach that is the focus of current research and development by the authors. The main functional elements of the IMS architecture are described, followed by thoughts regarding the practical implementation and the associated advantages (and potential disadvantages) that this would deliver. It is concluded that whilst an IMS-type approach would not represent a total replacement for conventional controls, it would represent an effective means to complement the protection already provided.

KEYWORDS

Intrusion Detection Systems, Intrusion Monitoring, User Profiling, Anomaly Detection.

INTRODUCTION

In typical IT systems, protection against unauthorised user activities is usually provided via login authentication and access controls. The majority of authentication schemes are based upon traditional password methods. The weaknesses of passwords are well-known (Jobusch and Oldehoeft, 1989), but their simplicity (from both user and developer perspectives) serves to ensure their continued use. A significant issue with passwords is that they typically provide a one-off authentication judgement at the beginning of a user session. From that point, protection against unauthorised user activity is reliant upon access controls applied to specific data and resources. Whilst these can be utilised in an effective manner, they are themselves reliant upon appropriate system administration to grant suitable access rights and privileges to users. However, depending upon the level of control imposed, this scenario still offers the potential for unauthorised activity. The normal means of monitoring and identifying this is via audit trails, which maintain a record of nominated security-relevant activities within the system and can be inspected at a later time in order to identify anomalies. The problem with this approach is that any detection of unauthorised activity will be retrospective, when significant damage may already have been done. In addition, previous research findings suggest that while many organisations may maintain audit trail information,

only a small percentage (10%) of them actively follow-up the information collected (Gliss, 1990). As such, breaches of security may potentially remain unnoticed for some time. What is, therefore, required is an automated, proactive means of detecting and responding to unauthorised access/activity. Such a solution is provided by intrusion detection systems.

The concept of intrusion detection can be traced to original work by Denning (1987), who proposed a model for an intrusion detection system (IDS). This work led to a number of practical IDS implementations from various organisations, in particular the IDES system from SRI (Lunt, 1990). Indeed, intrusion monitoring is still an area of active research[1], indicating that the overall issue has yet to be resolved.

This paper presents an approach to guard against these classes of intrusion and misuse, in the form of the Intrusion Monitoring System (IMS) architecture. The discussion begins by presenting the principles that underlie the intrusion monitoring process, followed by a conceptual description of the IMS architecture. The paper then proceeds to examine a number of issues relating to practical implementation aspects.

PRINCIPLES OF INTRUSION MONITORING

This section presents a number of general principles that underlie the concept of intrusion monitoring and detection, which will enable the architectural approach proposed by IMS to be more fully understood and appreciated.

Categorising system intrusions and misuse

At the highest level, intrusions or misuse will be the result of actions by *users* or *processes*, which will operate on one or more *targets* (which may include data (files), system devices and other users or processes). The purpose of introducing supervision will be two-fold:

1. to ensure that systems are only accessed by authorised users;
2. to ensure that systems are only used for authorised purposes.

User actions can be categorised as being either *legitimate* or *illegitimate.* However, it is useful if a more detailed breakdown than this can be derived for the different potential classes of illegitimate activity. For example, all of the following scenarios represent types of illegitimate activity that should be monitored:

- an illegitimate action that is still within the normal authorisation of a valid user (i.e. abuse of privileges);
- an action by a valid user which is outside the normal limits of authorisation;
- any action by an unauthorised user.

[1] See, for example, http://www.securitysearch.net/Research_and_Education/Intrusion_Detection_/

In addition, it is necessary to recognise differences in the types of potential system abuser. These have already been comprehensively categorised by Anderson (1980), and are described in table I.

Abuser Type	Description
External Penetrators	Outsiders attempting or gaining unauthorised access to the system.
Internal Penetrators	Authorised users of the system who access data, resources or programs to which they are not entitled. Sub-categorised into: • *Masqueraders* Users who operate under the identity of another user. • *Clandestine users* Users who evade access controls and auditing.
Misfeasors	Users who are authorised to use the system and resources accessed, but misuse their privileges.

Table I: Categories of system abuser

These groupings are considered appropriate for describing the different types of user-related abuse within an intrusion monitoring framework and will, therefore, be adopted for the remainder of the discussion. Whilst it is also possible to develop a deeper profile of potential intruders, by considering factors such as the common motivations behind abuse (e.g. money, ideology, egotism etc.), these are not explored here as knowledge of them would not contribute to the process of detection.

It should be noted that Anderson's categorisations do not take into account any of the categories of abuse that may result from software activity (e.g. viruses, Trojan Horses etc.). This is understandable given that the analysis was made in 1980 before such incidents had become commonplace. However, there has been a significant increase in such attacks over the last decade and evidence suggests that viruses are now the major cause of security breaches in both networked and standalone PC systems (National Computing Centre, 1998). It is now extremely unlikely that the problem will ever disappear and, therefore, countering such activity should also be within the scope of a comprehensive monitor. As a consequence, a further category of intrusion, called *malicious process* (or *malware*), can be added to Anderson's list. These may introduce various undesirable consequences, including the alteration or destruction of data, creation of false data, degradation of system performance, crashing of systems or other effects that might render data or systems inaccessible (Brunnstein et al. 1990).

Monitoring and detecting intrusions

The supervision of activities (and resulting anomaly detection) can be based upon user behaviour profiles and generic intrusion indicators. These approaches are common to other intrusion monitoring architectures, such as the IDES system mentioned earlier.

User profiles could conceivably hold a range of identification, authentication and behavioural information relating to registered users. Examples of potential profiled characteristics would include:

- system access times and locations;
- typical levels of system resource utilisation;
- application and file usage;
- methods of user interaction (e.g. GUI versus command line);
- biometric information (encompassing both physiological and behavioural characteristics).

The use of biometric monitoring is considered to be particularly appropriate to prevent impostor penetration and masquerade attacks. A number of options exist that could be employed in this context, including keystroke analysis (i.e. monitoring of the current user's typing style), face recognition and voice recognition (Miller, 1994).

It is also recognised that some classes of intrusion or misuse can be trapped without identifying departures from historical patterns of user behaviour. As such, generic intrusion rules (also known as attack signatures) may be utilised to identify the occurrence of events that are suspicious in themselves (i.e. irrespective of the user involved). Examples of such generic indicators would include the following:

- consecutive access violations;
- out of hours access;
- account overuse / simultaneous access;
- use of inactive accounts;
- copying of password file;
- extensive use of "help" systems;
- modification of an executable file.

While none of these alone would provide sufficient indication to state that an intrusion was in progress, the combination of two or more œould be considered more persuasive. In the IMS context, the occurrence of any such events would increase the alert status of the system (which, as discussed later, could result in a range of potential responses as different threshold values were reached).

A full IMS would operate by comparing current system activity against information held in a knowledge base. The knowledge base would effectively maintain two "models" of activity for reference by IMS:

- normal activity (i.e. the user behavioural profiles);
- intrusive activity (i.e. the generic rules).

These models will determine what types of activities and events the system will look for and, as such, an event will be judged to be indicative of a suspected intrusion if :

- it is *compatible* with *intrusive* activity ***OR***
- it is *incompatible* with *normal* activity.

Having considered these principles, the proposed architecture for a practical monitoring system can now be presented.

INTRUSION MONITORING SYSTEM ARCHITECTURE

At a high level, the IMS architecture is based upon the concept of a centralised *Host* handling the monitoring and supervision of one or more networked *Clients* running on local workstations. The purpose of the Clients is to collect the required data relating to user and process activity and respond to any suspected intrusions detected by the Host.

All behaviour profiles, generic rules and such like are 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 Host and Client systems will be comprised of a number of modules, each handling a different aspect of the overall intrusion monitoring task, as illustrated in figure 1. The modules shown are intended to represent the conceptual elements of the system, but could also equate to the coded functional elements in a full implementation. The key aspects of this design are defined in the sections that follow.

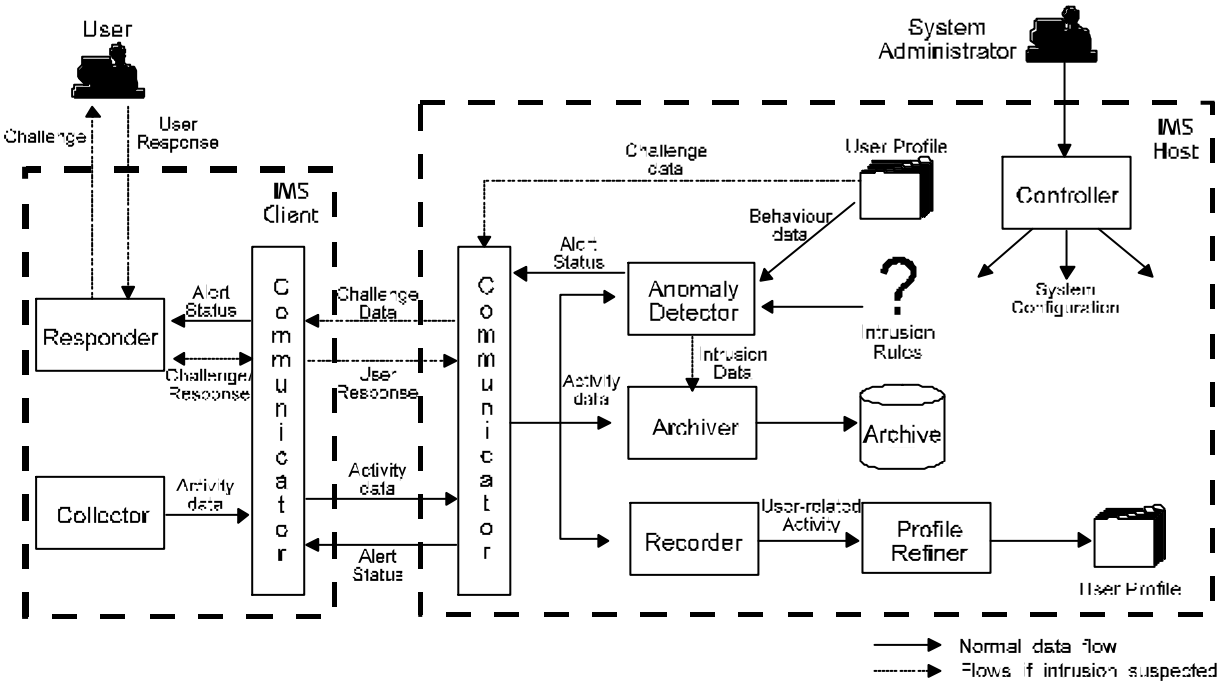

Figure 1: IMS Architecture

Anomaly Detector

The *Anomaly Detector* analyses user and process activity for signs of suspected intrusion, comparing it against the behaviour profiles (class and user-specific) that apply to the current user's (claimed) identity as well as against the generic intrusion rules. In practice, this module will be comprised of a number of further sub-modules, each handling a specific aspect of anomaly detection and behaviour monitoring (e.g. keystroke analysis).

The detector maintains an *alert status table*, with entries existing throughout the life of each user-initiated session or process to indicate the level of detected anomalies and thereby the confidence of a potential intrusion. Each entry contains the basic information shown in figure 2, which is examined and updated each time activity data relating to the relevant user / process is analysed.

User / Process ID	Alert Status Level	Idle Time	# previous challenges	Session start	# access violations

Figure 2: Structure of Alert Status table entry

It is envisaged that, at its most basic, the "alert status level" could be a simple aggregate value based on the number of behavioural anomalies detected and intrusion rules satisfied (with the monitored characteristics and rules having been weighted to indicate their significance). The entry relating to "idle time" will be used to allow the phased reduction of the alert status level after certain periods of inactivity. Recording a tally of previous challenges would then be used as a safeguard to determine whether the level of IMS response should be escalated in response to an anomaly even if the alert status is currently low (i.e. as a result of the phased reduction). As the figure illustrates, the table might also be used to store other information, such as the time of session / process initiation or the number of access violations incurred. These would be used for the purposes of on-going comparison against behaviour profiles (for example, session start time could be used to derive the current session length) and would also be required to be maintained throughout the live of the session. It should be noted that some of the table entries are most applicable in the context of monitoring user sessions and will be redundant in the case of process supervision.

The alert status level would increase in response to both departures from a user's historical behaviour profile and the satisfaction of generic intrusion indicators. Under normal circumstances, the detector would commence supervision of a session with an alert status of zero (i.e. no suspicion of an intrusion). However, factors such as failed login attempts, system configuration anomalies and the like could cause it to begin with a non-zero status so that it is essentially more sensitive to further anomalies in the initial instance. The alert status would be reduced after successful challenges or after a sufficient period of normal activity to allow the system to discount the previous anomaly.

Profile Refiner

It is desirable for IMS to utilise user-related activity data in two ways - to analyse for anomaly detection and as the basis for updating behaviour profiles. This second point recognises the possibility that user behaviour may legitimately alter over time (e.g. as a result of access to new applications, improvements in typing ability etc.). The purpose of the *Profile Refiner* would, therefore, be to provide an automatic means for user-specific profiles to be updated to account for such changes.

It would be most appropriate for the *Profile Refiner* to be based upon a neural network approach (Bishop, 1995), given that the inherent ability to analyse and recognise patterns could allow behavioural characteristics to be identified that might not be apparent to a human observer. In this way, the effectiveness of the system would have the potential to improve over time, in that it could gradually learn more patterns of legitimate activity for each user (building upon the foundation provided by the generic rules and the initial profiles). It might also be possible to determine which of the profiled characteristics provide the best discriminators for each user and thereby establish (for example) primary, secondary and tertiary level behaviour indicators (with the primary level representing the most reliable identity verifiers). This hierarchy could also be extended to allow for the fact that some characteristics may represent negative indicators (i.e. those that, despite refinement, are found to cause a high level of false alarms).

It would be undesirable for the *Profile Refiner* to utilise data that is later found to be anomalous. Refinement should, therefore, only take place after the termination of user sessions (provided, of course, that no intrusions were proven during this time). However, it is also considered sensible to allow refinement to proceed if any challenges that were generated were correctly answered by the user (the reason being that the generation of the alert may be indicative that legitimate behaviour has departed from the profile and that refinement is, therefore, necessary). However, in order to help guard against the recognised problem that misfeasors will answer challenges correctly, refinement should be performed on the proviso that the number of alerts raised was small *relative to the length of the session* (i.e. two alerts in a three hour session would be acceptable, whereas the same number in a ten minute session would be very suspicious). Additionally, any activity occurring during periods where supervision of the relevant aspect was suspended could not reliably be used for profile refinement.

User-specific profile records would also incorporate a series of flags to indicate whether the individual behaviour characteristics are ready to be used in supervision or still being developed. This will allow a gradual training period to be defined for new user profiles without the IMS continually generating intrusion alerts (the flags would also allow a specific "refinement only" period to be established for existing profiles that have proved to be inadequate for the legitimate user). The purpose of associating flags with each profile characteristic is so that some degree of monitoring could still continue whilst other aspects are being (re)trained. The flags could also be used to allow the total disablement of some aspects of monitoring if, for example, some characteristics are found to be inappropriate to certain users.

Data relating to *process* activity would not be used for refinement as the generic rulebase would remain static (unless specific information on new intrusion methods is introduced by the system administrator).

Recorder

The *Recorder* handles the short-term storage of user-related activity data during the period of a user session and focuses specifically upon the collection of data relating to the profiled characteristics of a given user (e.g. collection of keystroke data in relation to the typing profile). Upon termination, the information will be used as input to the *Profile Refiner*, provided that the session was not considered anomalous. In the event of a proven anomaly, the *Recorder* can discard its stored information for the session.

Archiver

The *Archiver* collects data relating to *all* system activity and stores 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 storage will occur regardless of whether sessions / processes are regarded as anomalous and details of all security relevant events will be archived. Such events will include login failures, intrusion alerts, authentication challenges, suspended sessions and the like. The basic format of the archive records would be as shown in figure 3.

Date	Time	User / Process ID	Logged Event	Privileges	Resources utilised

Figure 3: IMS Archive record structure

However, in order to conserve storage space, it may be desirable in some scenarios to only record details of certain types of event. The *Archiver* would, therefore, be configurable to suit the preferences of the establishment involved (note that the same would *not* necessarily be true for the *Recorder* as this would always need to collect information on any activities for which profile refinement may later occur). The long-term retention period of archived details would be determined by the security policy of the organisation involved.

Collector

The *Collector* represents the interface between the IMS and the existing information system / applications, with the responsibility for obtaining information on all relevant user and system activity. The module would be required to operate in such a way as to encompass, but be independent of, all system applications. It is envisaged that this could be best achieved by implementation at the operating system (OS) level, such that key events also lead to IMS notification. For example, a significant proportion of data collection could be based around the interception and redirection of selected OS interrupts and service requests (such as file input / output, application execution, keyboard input). These would be monitored with two objectives:

1. to collect data on those events which pertain to monitored behaviour characteristics;

2. to identify those events which may affect the security of the system (for comparison against generic intrusion indicators).

In some cases, the required data could be obtained directly from existing audit trail records on the underlying system. However, with certain aspects (e.g. keystroke analysis) the required information will not be maintained in audit trails and implementation may, therefore, require a significant number of operating system links. Whilst this would serve to make this aspect of IMS very system specific, it would be considerably more efficient than attempting to modify each individual application to specifically provide relevant information to IMS. The system specific coding of the *Collector* would only need to be done once, whereas modifications would be required to all current and future applications (which would be likely to be a non-trivial undertaking and potentially impossible in the case of commercial packages where source code may be unobtainable).

As with the configuration of the *Archiver*, the resolution of data collection would be determined at the Host by the System Administrator.

Responder

This module resides in the IMS 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 issue of a user authentication challenge, suspension of a session or cancellation of a process. The issue of appropriate response is discussed in more detail later in the paper.

In some implementation scenarios, the *Responder* might also be responsible for handling the initial user identification and authentication process that is required to gain access to the system in the first instance.

Communicator

The *Communicator* provides the network communications interface between the Host and the Client(s) operating on the local systems. As such, the functionality of this module is duplicated on both sides of the link. The principal functions would include transmitting user and process information to the Host and then subsequently keeping the Client(s) informed of the current alert status. If implemented in a heterogeneous environment, the Client side of the module would be responsible for resolving any operating system differences that exist within the monitoring domain so that information could be presented to the Host in a consistent, standardised format.

Controller

This module is provided for use by the System Administrator to allow the operation of the IMS system to be configured. On the Host side, such configuration would apply to the following modules :

- *Anomaly Detector*, e.g. behaviour characteristics to consider / prioritise, generic rules in operation;
- *Profile Refiner*, e.g. frequency of refinement, acceptable thresholds for challenges within a session;
- *Archiver*, e.g. level of detail required, specific events to record or exclude from logging.

For the Client side, the operation of the following modules would be controlled:

- *Collector*, e.g. the level of data collection (linked to the characteristics being monitored by the *Anomaly Detector*).
- *Responder*, e.g. the level of response required at each alert status level.

These settings would obviously be controlled and recorded through the Host system. The configuration of the local Client(s) would then be established at the time of session initiation.

In addition to the above, several other features would also be provided under the auspices of the *Controller* module. These would include facilities such as user profile management, update of the generic rulebase and the like.

User profiles

IMS profiles could conceivably hold a range of identification, authentication and behavioural information relating to legitimate users. The profiles would use a number of methods to represent measures of user behaviour:

- frequency tables (e.g. for file access);
- means and standard deviations (e.g. for keystroke / typing profiles);
- ranges (e.g. valid access times);
- lists (e.g. for valid access locations).
- a combination of methods (e.g. a list of valid access locations which also indicate the relative frequency of use).

The profile data obviously requires secure storage to prevent unauthorised browsing or tampering by potential impostors. If users were able to modify profile information it would be possible for them to adjust the records of other users to match their own (and, therefore, allow them to access the account in place of the legitimate owner). Whilst disclosure of the profile statistics may not initially appear to pose such a threat, it could still be a problem in the case of a *determined* impostor. For example, if the characteristics of the 'target' user were known, the impostor would have a concrete statement of what he / she would be

required to mimic. An alternative option would, of course, be to subsequently enlist the help of an accomplice with a comparable profile. At the very least, this dictates a requirement for encrypted storage, as used with the password files in the majority of commercial operating systems (Morris and Thompson, 1978).

ISSUES RELATED TO INTRUSION MONITORING

This section presents further discussion of a number of the issues that were mentioned during the description of the IMS modules. The issues in question are the restriction of user activities, suspension of supervision and types of response to suspected intrusion.

Restriction of user activities

It is considered feasible for the alert status level to be inter-linked with the types of activity that a subject is allowed to perform, such that a phased reduction of permitted behaviour would occur as the level increases. In this way, highly sensitive activities and / or information could be denied if there is any doubt over the current user's legitimacy, whilst still allowing more mundane activities to continue. The approach would demand that a maximum alert status threshold be associated with each of the activities or objects that the IMS is to control. If the current status level was then to exceed this, the activity or object would become unavailable. For example, consider the thresholds in table II associated with two objects (*wordprocessor* and *database*) and the activities *create* and *delete file*. If the current alert status level were 5 then the user would not be permitted to access the database or to perform any file deletion. However, the creation of a file using the wordprocessor application would still be possible.

Activity / Object	Alert Status Threshold
Wordprocessor	8
Database	2
Create file	8
Delete file	3

Table II: Alert status threshold table

Such a threshold table would be maintained within IMS, but the values would initially need to be assigned (and, if necessary, subsequently updated) by the system administrator. It must be said that the potential for error would make this approach inappropriate in many scenarios (for example, the denial of data access in sensitive applications could be most unwelcome). In any case, it would be advisable for the system administrator to be notified whenever behaviour restrictions were being imposed so that the situation could be investigated (in case legitimate users were being unintentionally impeded).

Suspension of supervision

In some cases it is envisaged that continuous behaviour monitoring at *all* times throughout a user session may not be strictly necessary or even advantageous. This is especially true in the case of the mechanisms aimed solely at the detection of penetrators (e.g. keystroke analysis). The rationale here is that, after a reasonable amount of uninterrupted behaviour analysis (i.e. with no challenges and no significant periods of user inactivity), the monitoring system should have been able to accurately determine the legitimacy of the current user (e.g. previous research has indicated that, using keystroke analysis, a reliable authentication judgement should be obtainable within 400 keystrokes in a real-time monitoring context (Furnell, 1995)). If an impostor is not suspected at this point then it is extremely unlikely that further monitoring will detect one (indeed, monitoring for longer than is necessary would simply allow more opportunity for false rejections to occur and place an additional load on the system). In view of this, it is considered that monitoring activity during the following periods is likely to be most crucial in terms of impostor detection (with supervision being temporarily suspended at other times):

- during the period immediately after the start of the session (when the authenticity of the user has yet to be conclusively proven);
- during the time after any significant period of inactivity (during which an impostor could potentially have replaced the legitimate user).

Important considerations here would obviously be the period of monitoring necessary before suspension of supervision and also what length of time would constitute the "significant period of inactivity" necessary for it to be resumed. Suggested periods would depend upon the monitored characteristics (e.g. monitoring of keystroke dynamics could yield an authentication judgement more quickly than monitoring of application usage), but a general rule could be to monitor up to five minutes of non-anomalous activity before suspension in order to allow a sufficient appraisal of the user to be made. Approximately 2-3 minutes of inactivity would then be seen as a suitable trigger for monitoring to resume, as this length of time could have allowed sufficient opportunity for impostor intervention. In a practical implementation, both of these aspects would be configurable so that the optimum levels could be established.

It should be noted that this approach would *not* be adequate for detection of misfeasor activity, as this could very well proceed *after* authentication has been established. Therefore, if suspension of monitoring was still to be incorporated, it would be sensible to periodically reintroduce supervision at *random* intervals as an additional safeguard (this would also help to guard against a situation where an impostor / penetrator might be able to replace the authorised user without there being a significant period of inactivity).

This idea is primarily suggested as a means of minimising the likelihood of false rejections in the practical context. However, a further advantage in the context of practical implementation would be that it would reduce the significant processing overhead that would be associated with continuous monitoring in an environment with a large number of Client machines

Response to suspected intrusions

The existence and operation of IMS should ideally remain transparent to the user unless an anomaly is suspected. As previously stated, a suspected intrusion will cause IMS to automatically perform some further action (the nature of which will vary depending upon the type of intrusion involved). Options here include:

- issuing of an explicit request (or challenge) for further authentication;
- recording of details in an intrusion log for later inspection / investigation;
- immediate notification of the system manager (i.e. an intrusion alarm);
- phased reduction of permitted behaviour;
- locking of the intruder's terminal;
- termination (or suspension) of the anomalous session / process.

The degree of automatic response is an important consideration and, as indicated above, must be matched to the severity of the suspected intrusion. For example, if there is high confidence that an activity represents an intrusion or if a particularly serious breach is suspected, then the maximum countermeasure response should result. However, in lesser scenarios more limited responses will be appropriate (e.g. simply writing details to the intrusion log).

There is an obvious danger that any option which allows the user to continue working whilst the anomaly is investigated would also allow more time for an intruder to cause damage. At the other extreme it would be undesirable for the system to terminate a session or process without a *very* high degree of certainty that an intrusion was in progress. Therefore, the first two options above are considered to be the most appropriate as initial forms of response.

In practice, there are several possibilities for the type of challenge that the system could issue in the event of a suspected intrusion. The original system password would obviously be inadequate, given that it may have already been compromised in order for an intruder to have gained access in the first place. It is desirable that the challenge be such that it allows any legitimate user to resume work quickly with minimal interruption (i.e. it should be easy for them to overcome, whilst still trapping impostors). A suggestion is that a (short) series of question and answer type challenges be posed to the user (Haga and Zviran, 1991), who would then need to answer them correctly in order to proceed further. These could be based upon cognitive and / or associative information, with valid responses having been obtained and stored in conjunction with the original user profiling. If several (e.g. 5 to 10) such questions were to be obtained from users during profiling then the challenge could be based upon a random selection from the set (further reducing the chance of impostors being able to compromise the system).

There are, however, a number of scenarios in which this approach would be ineffective. Firstly, it must be remembered that any form of "authentication-based" challenge would be an inadequate countermeasure against misfeasors. They would obviously be able to respond correctly to such challenges (having originally supplied the information themselves)

and then continue with unauthorised activity. There is a solution here in the realisation that continuing anomalies would lead to a succession of intrusion alerts; an event that would be suspicious in itself. At this point, the IMS response could then change to a method that would effectively combat misfeasors as well (e.g. a session lock or a trigger for system manager investigation). Nevertheless, this would still enable misfeasors to continue for longer than other classes of intruder (albeit with intermediate challenge(s)) before the system locks them out.

A second problem / exception relates to suspected malicious processes - these cannot be issued with a challenge to which they may respond and verify their legitimacy. This in turn places more importance on the correctness of the resulting IMS response (e.g. the dangers of suspending / deleting a legitimate, and possibly essential, process or failing to take positive action against a genuinely destructive one).

Finally, some classes of anomaly (for example, login failures based on unrecognised user identities) cannot be tied to a specific user and, as such, the issue of a challenge based upon profile information is again inappropriate. However, it is conceivable that some form of generic challenge could be issued (the answer to which would be known by legitimate system users), with invalid responses causing the IMS to proceed to its next level of countermeasure (e.g. system manager notification, terminal lockout).

IMS IMPLEMENTATION ISSUES

The IMS concept is considered most appropriate to implementation in a networked environment, for the following reasons:

- standalone systems will most often be dedicated to a single user. As such, more traditional authentication and access controls (e.g. passwords) will probably be sufficient to ensure security if they are correctly implemented.

- implementation of a full IMS would be likely to degrade the performance of a standalone system.

- networked systems provide more potential for collecting monitoring information. Many statistics (e.g. access location, resource usage) would not be appropriate to a standalone environment.

In this scenario, the Host would be centralised with multiple IMS Clients being used to monitor activity on the individual workstations. The purpose of the Clients would be to collect any activity data that is generated locally (e.g. keystroke timings) and to enforce IMS restrictions in suspected intrusion scenarios (e.g. issue a challenge, lock access to the system etc.). In such a scenario it would be necessary to maintain the security of the IMS Clients on the individual machines to ensure that their operation cannot be compromised (e.g. by a malicious user trying to avoid detection).

The realisation of the IMS approach is considered to have a number of advantages, as listed below.

- *Improved security*
  This is advantageous in any information system, and is achieved here due to the continuous nature of supervision. User authentication is no longer restricted to the discrete judgement(s) possible with passwords and misuse will be identifiable much earlier than with traditional auditing. In addition, the fact that much of the supervision is based upon behavioural characteristics makes it more difficult for users themselves to undermine security (e.g. by allowing colleagues unauthorised access to their accounts) as they cannot easily transfer these abilities to other users.

- *Cost*
  Advantages here result from the fact that it is possible to implement the concept entirely in software at the user end, whereas many frequently suggested authentication enhancement schemes (e.g. smart cards, other biometric methods) are reliant upon specialised equipment at each user workstation. This makes the technique particularly suited to financially constrained environments.

- *Convenience*
  This comes from the fact that the supervision can be performed transparently, in a non-intrusive manner. In addition, the fact that the IMS would demand nothing special from the users (e.g. they are not required to remember additional password-type information or possess any physical token) means that its operation should not undermine the existing staff culture, which is recognised as an important issue in the introduction of security (Warren et al. 1995).

There are also a number of inherent disadvantages in the concept of IMS (and, indeed, any other type of comprehensive monitoring and supervision system). The principal concerns are highlighted below.

- The operation of IMS Clients and/or data collection will consume system resources and may degrade overall performance. The collection of detailed audit trail data typically degrades machine performance by between 5 and 20 percent (Wolfe, 1992; Mukherjee et al. 1994). An IMS performing full behavioural monitoring and testing of generic intrusion rules would be envisaged to introduce a similar burden.

- Transmission of data from Clients to the Host will result in a loss of network bandwidth and a loss of timeliness of data.

- Maintenance of the IMS itself would entail a more significant management / administration burden in the affected systems. For example, correcting problems with behaviour profiles would be a more complex operation than cancelling a forgotten password. At the same time, however, other duties (such as inspection of audit trails) would be reduced, so the new demands would at least be somewhat offset.

- The overall concept of continuous supervision raises a question of user acceptance. It is conceivable that there may be mistrust and resentment of the system on the grounds of it being seen as a means of monitoring legitimate work and staff performance as opposed to just guarding against intruders. It would, therefore, be important to ensure that the system is perceived as a "Caring Mother" rather than a "Big Brother".

In general terms, the likely advantages when compared to other means of protection are considered sufficient to outweigh these points. In view of this, the authors are currently developing an implementation of the IMS approach for the Windows NT environment (Dowland and Furnell, 2000).

CONCLUSIONS

The paper has described the concept of intrusion monitoring and a potential approach by which it may be realised in modern networked systems. The IMS concept is not intended as a total replacement for conventional authentication and access control methods (although in some cases it will offer an opportunity for more dated approaches to be replaced). In the majority of systems, supervision could be incorporated alongside other methods to complement the security already provided. In addition, it will have little or no effect upon the need for physical security and personnel-related measures within an organisation. There are also some important aspects of logical security that are not addressed (e.g. protection of data communications) which further highlight the potential need for a wider IT security framework.

REFERENCES

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

Bishop, C.M. (1995), *Neural Networks for Pattern Recognition*. Oxford University Press.

Brunnstein, K; Fischer-Hubner, S.; and Swimmer, M. (1990), "Classification of Computer Anomalies", In *Proceedings of 13th National Computer Security Conference* (Washington DC, USA, Oct.1-4), pp374-384.

Denning, D.E. (1987), "An intrusion-detection model", *IEEE Transactions on Software Engineering*, SE-13(2), pp222-232.

Dowland, P.S. and Furnell, S.M. (2000), "Enhancing Operating System Authentication Techniques", to appear in *Proceedings of the Second International Network Conference (INC 2000)* (Plymouth, UK, 3-6 July).

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

Gliss, H. (1990), "A survey of computer abuse", in *Proceedings of Compsec 90 International* (London, UK, Oct. 10-12), pp495-517.

Haga, W.J. and Zviran, M. (1991), "Question-and-Answer Passwords: An Empirical Evalution", *Information Systems*, Vol. 16, No.3, pp335-343.

Jobusch, D.L. and Oldehoeft, A.E. (1989), "A Survey of Password Mechanisms : Part 1", *Computers & Security*, Vol. 8, No. 7, pp587-604.

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

Miller, B. (1994), "Vital signs of identity", *IEEE Spectrum*, February 1994.

Morris, R. and Thompson, K. 1978. "Password Security: A Case History", In *UNIX Time-Sharing System: UNIX Programmer's Manual, Seventh Edition, Volume 2*. Bell Laboratories (1983): 595-601.

Mukherjee, B.; Heberlein, L.T.; Levitt, K.N. (1994), "Network Intrusion Detection", *IEEE Networks*, Vol. 8, No. 3, pp26-41.

National Computing Centre. (1998), *BISS '98 – Inforamtion Security. The true cost to Business*. National Computing Centre Limited, Manchester, UK.

Warren, M.J, Sanders, P.W. and Gaunt, P.N. (1995), "Participational Management and the Implementation of Multimedia Systems", in *Proceedings of MEDIACOMM 95 - International Conference on Multimedia Communications* (Southampton, UK, 11-12 April), pp131-135.

Wolfe, A.D. (1992), "Securing the distributed environment: a question of trust", *Patricia Seybold's Network Monitor*, Vol. 7, No. 1, pp3-19.