Proceedings of the 2004 IEEE
Workshop on Information Assurance and Security
United States Military Academy, West Point, NY, 10–11 June 2004

# Molehunt: Near-line Semantic Activity Tracing

Stephen D. Wolthusen
Fraunhofer-IGD, Germany
wolt@igd.fhg.de

*Abstract*— **This paper discusses threats posed by low granularity in access to confidential (classified) data typically found at lower protection levels, namely direct access beyond need to know and the correlation of materials yielding more sensitive aggregate data by both insider threats and malware, an area of particular concern for intelligence analysis. It is argued that while active security controls at both the procedural and technical level are currently not pragmatically feasible, near-line semantic monitoring particularly at the file system but also at the network level can provide capabilities to detect anomalous and also directed malicious activity. A mechanism for implementing the tracing and monitoring mechanism on an COTS operating system is described.**

## I. Introduction

Classification mechanisms for information and user clearance along with compartmentalization are highly effective means for enforcing confidentiality policies. They also, however, incur significant direct cost for the administrative overhead required to perform classification and declassification operations as well as indirect costs resulting from limited productivity and friction due to excessive restrictions on the need to know.

The most appealing information system to support these mechanisms, namely multi-level secure (MLS) information systems, represent a significant risk in that even very subtle flaws in their design, implementation, and operation can compromise their security properties. Given the difficulties inherent in the design and development of suitably high assurance systems, there are few MLS systems in existence, particularly not as commercial off the shelf (COTS) systems. Even for such MLS systems, however, availability of desirable or required application programs can be problematic. A common approach to these problems in handling material whose confidentiality must be protected is therefore to operate at lower protection levels [1] (i.e. dedicated, system high, or compartmented modes), an approach that are also far more common in commercial environments where the added overhead of higher protection levels can rarely be justified.

This approach and its reliance on procedural and environmental protection measures and controls, however, does not fully address a number of threats. In the following, section II outlines these threats. An approach for countering the threats is described in section III, followed by a description of the sensor mechanisms employed to obtain the requisite data for analysis in section IV. The near-line analysis mechanisms and heuristics are described in section VI. Section VIII subsequently discusses related approaches and fields; section IX provides a summary of results obtained thus far and a discussion of ongoing and future research.

## II. Threat Analysis

The pragmatically achievable limits in the granularity of compartmentalization imply that several situations, frequently distinguishable only at the procedural level, can arise. These can be further subdivided into the agents involved, namely humans or malware.

The type of threat this paper is concerned with is best illustrated by the case of Robert P. Hanssen, an agent of the U.S. Federal Bureau of Investigation. Hanssen had joined the FBI in 1976 but was recruited by the USSR's GRU (later KGB) in 1979. Subsequently, Hanssen became the Chief of the National Security Threat List (NSTL) Unit at FBI Headquarters in 1992.

While Hanssen was removed from this position in 1994 in part in response to security breaches involving illicit access to Soviet counterintelligence data from the FBI's National Security Division, he continued to have access to highly sensitive data, including the FBI's Automated Case Support (ACS) information system, in the position of liaison to the U.S. State Department's Office of Foreign Missions (OFM) until his arrest in 2001. Hanssen found and subsequently sought out information pertaining to his own case from the ACS during this time[1] [2], [3], [4]. Beyond such individual events, low level dissemination of material (whether in intelligence analysis or in other areas concerned with sensitive information, e.g. in the financial services industry) that remains undetected or cannot be traced back can prove to be highly problematic and can yield valuable actionable intelligence to adversaries [5].

[1]The uploading of files related to this investigation were later determined to be procedural errors primarily due to failures in training.

### A. Insider Threats

Individuals may have access to documents in excess of their actual need to know under several circumstances. In case of multi-level secure and compartmented environments, the level of compartmentalization may be of insufficient granularity to fully differentiate need to know since procedural overhead is deemed to be excessive by the designating authority. At lower protection levels, need to know and formal access approval may also not coincide and, although no compartmentalization exists, operating on documents not directly related to current tasking can be considered inappropriate behavior.

Both types of access are formally authorized and therefore must be permitted by a technical security control enforcing the pertinent security policy semantics determined by the classification and clearance mechanism as bounded by the capabilities of available security controls. They may, however, be inappropriate at higher, mission-based semantic levels. Access to documents outside current mission and tasking scope is the most direct form of such inappropriate access, although e.g. intelligence analysts may legitimately require such broad cross-cutting information sources. Beyond such direct access, however, limited granularity of control also allows the aggregation and correlation of multiple sources (including open sources and sensitive but unclassified material); an activity that is even less amenable to enforcement by (technical) security controls than direct access.

Such aggregation and correlation activities are obviously fundamental e.g. to legitimate intelligence analysis and indeed represent the plurality of actual intelligence (as opposed to source materials) [6]. Moreover, even a far-ranging inquiry across seemingly unrelated areas may be highly desirable in extracting intelligence from disparate sources. A key characteristic of this type of threat is therefore that the appropriateness of a given pattern of access to documents can only be determined post factum — and even then the assessment may not be conclusive since the plausibility of the justifiability of a given access pattern ultimately rests on individual circumstances and personal judgment.

### B. Malware Threats

Particularly for systems operating at lower protection levels where no adequate technical security controls exist, the problems described in section II-A are further exacerbated by a vulnerability of most COTS operating systems used in such information systems to being compromised by malware (e.g. viruses, worms, Trojan horses) that can be introduced even into nominally isolated systems, e.g. during document transmission in certain word processing formats. Possible damage routines that can be deployed by such malware are scans for pertinent information (common malware for non-targeted use e.g. may extract credit card numbers and personal information subsequently used for fraud and identity theft); given the lack of controls[2] described in section II-A, this implies that the malware has full access to all information a legitimate individual has because it can assume the identity of that individual (i.e. a process with identical credentials). Besides problems of accountability (the assumption of presence of malware has been used to establish plausible deniability in court), these damage routines can easily be retargeted in conjunction with custom-designed malware intended to circumvent malware detection software.

Targeted malwares (which may in part be derived from readily existing malware) must be assumed to be undetectable by signature-based detection systems since, by definition, it has not been released and therefore observed before. This reduces the problem of detecting such malware to behavioral anomalies as in the case of section II-A, but may introduce additional complications e.g. if scanning and retrieval are obfuscated by randomizing the scan processes and interspersing targeted with non-targeted scan activities.

## III. Molehunt System Outline

Both security controls and analytical capabilities differ significantly depending on the filing mechanims used for source materials. Database management systems (particularly relational database management systems) are frequently more amenable to such controls. However, significant amounts of data stored and operated on are unstructured, organized in a variety of formats, are stored in file systems (where they may be accessed as files or possibly through an online retrieval system), and may contain multiple media types.

This paper therefore focuses on mechanisms for the interception and subsequent analysis of unstructured information as found in file systems and, to a lesser extent, online retrieval systems. Moreover, while the overall architecture described is applicable to most COTS operating systems, the requisite sensor components necessarily are heavily dependent on the operating system used. Without loss of generality this paper therefore focuses on the Microsoft Windows NT family of operating systems (abbreviated Windows NT in the following, but including the NT, 2000, XP, and 2003 releasese).

The system is divided into four components:

- A suite of sensor components that are embedded in the host operating system that can intercept and monitor all data retrieval from all file systems (both local and via network file systems) and selected network protocols (primarily HTTP and HTTP over TLS).
- A sensor data extraction stack (SDES) to analyze individual files and data streams for known file and media types and, where possible, to extract pertinent information into a normalized format for further processing by the anomaly detection and semantic modeling components.

- An anomaly detection mechanism operating both on the metadata level and the data extracted by the SDES mechanism.
- A semantic modeling module operating primarily on data extracted by the SDES mechanism to correlate and classify concepts analyzed by a given entity and behavioral patterns exhibited in the process of analysis and data retrieval.

The objective of this architecture is the ability to have a fully transparent mechanism for observing any access to file systems and network data traffic that cannot be bypassed by application programs and, at the same time, also does not require modification of user or application program behavior. The data to be captured by this interception mechanism consists of both metadata (e.g. file names and locations, timestamps, and access modes in case of files and source and destination addresses for network traffic) and the actual data retrieved.

Since this data would prove quite voluminous, the extent of interception must be limited based on the maximum acceptable degradation in response behavior during operation, the storage and transmission capacity available for intercepted data, and the capabilities and performance of the mechanisms subsequently available for analysis. Particularly the latter tradeoffs are, however, hard to assess quantitatively since, as noted in section II-A, the need for further analysis or data to corroborate a given hypothesis will frequently arise only after the fact.

The system architecture further assumes that the systems on which it is to be deployed are in a benign threat environment and administered by trusted personnel. This permits that significant portions of analytical processes can be performed on the monitored systems themselves since the total computational capabilities of even modest current personal computers significantly exceed those required by the user except during brief periods of intense activities. As a result of this excess capacity, the system is designed to perform only the actual interception, pre-filtering, and storage of data synchronous with the operations of the user (or application program); the extraction, analytical, and reporting steps are deferred as necessary to limit the degradation in response time experienced by the user.

Moreover, as discussed in section IV, only the interception, pre-filtering, and storage must be performed at the operating system kernel level; all other steps can be performed at the user level (albeit protected from all user processes), thereby significantly reducing the total developmental effort required. The requirement for a benign threat environment is exists since otherwise tampering (e.g. initiating a hardware reset of the system) or outright destruction of the computer can erase or modify data required for analytical processes. While incidents such as power failures or software flaws can induce loss of data in any case, the availability of additional computational power must be considered a significant factor in the trade-off.

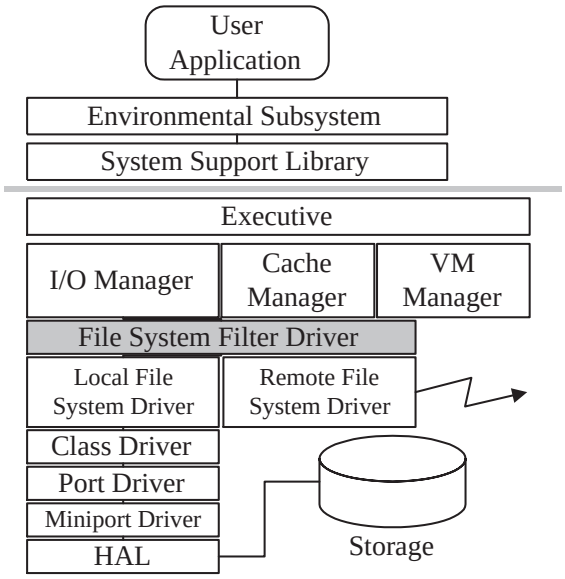


Fig. 1. File system interception mechanism for the Microsoft Windows NT family of operating systems

## IV. Sensor Components

The sensor components embedded in the operating system can be separated into file system and network traffic areas. However, since network file systems also must be considered, a naive implementation would require duplicate effort. To avoid this dupliaction, communication between the two sensor subsystems is required in which the file system sensor component notifies the network device sensor component of data streams it is already intercepting.

### *A. File System Sensors*

For the file system sensor, a component must be inserted into the operating system in such a way that it is non-bypassable and also non-removable by unprivileged users. This can be accomplished by inserting a module into the operating system kernel as described below.

While the APIs exposed via environmental subsystems [7], [8][2] by Windows NT are largely procedural in nature, the internal processing is asynchronous and packet-based. In this regard, Windows NT shares more with systems OpenVMS [9] than with Unix [10], although one major difference to OpenVMS is that, like Unix System V Release 4 and later derivatives, it has a unified file system cache and virtual memory architecture. The asynchronous mode of operation, however, hsa significant effects on how the interception of file accesses can be achieved.

The central component in the I/O architecture for the purposes of file system interception is the I/O manager. It creates I/O request packets (IRP) from incoming re-

[2]The actual subsystems vary; Windows XP and 2003 no longer include the IBM OS/2 subsystem.

quests[3] and ensures that all drivers for which an IRP is relevant are called with the IRP in the proper sequence. Each IRP sent to a kernel-mode driver represents a pending I/O request to that driver. An IRP will continue to be outstanding until the recipient of the IRP invokes the `IoCompleteRequest()` service routine for that particular IRP. Invoking `IoCompleteRequest()` on an IRP results in that I/O operation being marked as completed, and the I/O Manager then triggers any post-completion processing that was awaiting completion of the I/O request. Each request must be completed exactly once.

This mechanism lends itself to a layered processing approach in which IRPs are cascaded across several driver layers (possibly with additional IRPs created along the way at lower levels). As a side effect of this architecture, one can alter the functionality of the operating system by interposing additional layers in the driver stack. One example of such an interposition (an upper level file system filter driver) is shown in figure 1; for a more detailed exposition of the mechanisms involved and possible alterante choices see [11], [12], [13], [14], [15], [16]. The placement of the filtering (interception) layer as shown in figure 1 has the advantage of such a module being able to intercept generic (file-system independent) operations from upper operating system layers; this type of filter is called an (upper level) file system filter driver. Most importantly, the depicted interposition layer allows interceptions of all operations pertinent to individual files.

A number of special cases for handling exist (discussed in detail in an earlier paper [11]); of particular concern here, however, are network file system accesses. These are handled differently internally by Windows NT in that a special remote file system driver exists for network (redirector) file systems that, while nominally supporting the same interfaces as local file systems, requires specific handling of its interface semantics. This requirement for special handling (e.g. interactions with cache subsystem behavior) makes the additional overhead for notifying the network sensor mechanism described in section IV-B negligable.

Compared to the objectives discussed in [11], the interception discussed does not require changes to the intercepted file system itself, and therefore need not be concerned with maintaining cache consistency and file size. Instead, only metadata and actual file contents for selected files and file systems are recorded to a separate isolated storage location invisible to user processes.

Metadata is primarily obtained from interception of `IRP_MJ_CREATE` requests. This IRP is always issued when a file is accessed for the first time (not just for file creation) by an upper level function of a process. The sensor driver can subsequently issue a number of additional IRPs to gather all requisite information (e.g. `IRP_MJ_QUERY_INFORMATION`) and then record this information[4] in the database (cf. section IV-A.1). This information includes the process (user) accessing the file, its canonical location within the file system, timestamps for access modes (reading and writing), and the type of access desired.

The time of first access is also used to determine whether the metadata (and subsequently also the actual file content) is to be recorded at all. This can be configured dynamically by the security administrator through the use of configuration files that are cached at the kernel level. The reason for this mechanism (compared to e.g. retaining data in the system's Registry database) is primarily the performance impact of switching between kernel and user modes as well as limitations on such switches depending on the type and status of a given IRP and possible conflicts in accessing the Registry with other (user) processes.

Changes in configuration are therefore to be effected only through an explicit signal to the filter driver advising it to refresh the configuration at the earliest possible time. A second deselection criterion for metadata is caused by the fact that Windows NT uses the same IRP not only for files but also for a large variety of other objects including transient entities such as named pipes and shared memory segments. Since these typically do not share file semantics and may be performance sensitive, they should be omitted from interception.

All file systems of interest, including dynamically loaded file systems, must be intercepted; to this end, a notification callback for the file system minor functions `LoadFS` and `MountVolume` are used. The only exception to this mechanism, the RAW file system, is generally blocked for regular user processes since it permits circumvention of operating system access control mechanisms. Once the interception mechanism has determined that a file is accessed by a process of interest, metadata is stored in a database file with fixed record size indexed by the canonical file name (which must also include file system specific structures such as alternate data streams supported by NTFS), subject identity, and last access (indices are retained in main memory).

Subsequent read access for these files is then augmented by a write process that maps each paging read occurring for a write operation into a write paging request for the storage subsystem. This is achieved by performing a memory mapping between the respective uses, resulting in efficient caching and a reduced number of actual disk operations incurred (i.e. at most one paging write operation occurs for an arbitrary number of paging read operations provided that no modification of the pages read occurs on the part of the reading process).

The result (see section IV-A.1) is a snapshot of all pages read by the process. However, since the analytical processes described in sections V through VI require not only the actual pages read but also context to re-establish the

[3] With the exception of Fast I/O which bypasses this step, loosely patterned after the OpenVMS concept by the same name

semantics of file contents, backfilling of pages unread by the user process is required. This is accomplished by creating a worker thread that replicates the unread pages by initiating paging reads itself independent of the user process behavior. The computational complexity and particular the I/O load of this operation can be reduced significantly by utilizing the fact that the Cache Manager will perform a predictive read-ahead itself. By re-using the pages already cached and inducing predictive read-ahead, both total I/O operations and cache memory allocation can be minimized – however, there is a significant cost incurred in the additional paging writes that must be scheduled.

### A.1 File System Sensor Storage

The objective of the storage component for the file system mechanism is the minimization of required operations while executing I/O operations, resulting in a somewhat inefficient storage layout. Both metadata and individual files are allocated as sparse files[4], resulting in significant savings in storage required. The storage can be allocated on any file system, but to retain the invisibility of the Molehunt system to the user, this is best achieved by allocating a separate file system to it and hiding it (at the level of the file system filter driver) from the remainder of the operating system and hence from both observation and manipulation by users and application programs.

Metadata is stored as a single page (4 kB) per access record, indexed by the canonical file name, subject identity, and access time; only one record is written for a given process indicating the first time a file is being accessed (the actual index is built in a user space process). The metadata record also contains a reference to the location of the copy of the individual file. This location (file name) is the full canonical name of the original file, relocated to the storage file system (i.e. with an additional prefix identifying the storage file system).

## *B. Network Sensors*

In addition to traditional file system-based operations, information retrieval from network-based databases and services are also of particular interest. Using additional network-based sensors, this additional source of information can also be subjected to surveillance and subsequent analysis. Although there exists a large number of network as well as application protocols for such retrieval systems, the Internet Protocol and HTTP (and HTTPS) protocols clearly dominate as network and application protocol, respectively.

The mechanisms for interposition of an interception mechanism (with additional transparent in-line proxying for HTTPS (TLS) connections) have been described earlier[5] [17], [18]; by inserting kernel modules and driver components at several locations within the Windows NT network protocol stack, all inbound and outbound network traffic can be observed transparently without affecting application programs.

For the purposes of this paper, only a subset of the entire network traffic is of interest[5]; particularly HTML and XML data as well as multimedia documents. Based on the intercepted HTTP protocol elements as well as layered protocols (particularly MIME encodings and file format information), the interception layer can extract individual file elements (e.g. HTML text) from the network data stream and forward these cached elements to the storage mechanism described in section IV-A.1.

# V. Sensor Data Extraction

Sensor data extraction occurs in multiple steps on demand from one of the detection mechanisms (which may cache the extracted information separately); all of which are performed by a system service operating in the background that does not communicate with regular user processes but can trigger file backfilling in the file system filter driver through IOCTL calls (in case the caching mechanism has not yet backfilled a file selected for data extraction). The first extraction step is the identification of possible outer encodings (e.g. file compression, MIME transport encodings), followed by the determination of file type; this is performed heuristically by analyzing the start of the file for either explicit file type information or sufficient data to deduce file type. Based on the file type classification, files are forwarded through an extensible dispatcher system to media-specific extractors. These extractors are initially defined for text, image, and audio data.

## *A. Text Data*

For text data – as for all other media types – there exists a large number of file formats and encodings of which pragmatically only a selection can be addressed. Other than for plain text (for which the encoding may still need to be determined if data is not presented in ISO 646 or 10646 format), a translation filter is still required. These filters can (partially) parse markup languages such as SGML, XML, and HTML, although in the latter case extraction is limited to removing markup language since attributes cannot be extracted reliably as is the case for SGML and XML.

Of particular interest for XML document decoding is the OpenOffice document type descriptor (the SXW format). To avoid custom development of complex filters that need to be adapted frequently to version updates, proprietary

[4]This feature is available only beginning with Microsoft Windows 2000 in version 5.0 of NTFS and up; it also requires that the sensor storage is placed on an NTFS file system, which, however, is also highly desirable for security reasons alone.

[5]As with files, superencoding and superencipherment as well as the use of different protocols can thwart this interception; however, such anomalies are also likely to be remarked upon by a network intrusion detection system.

text formats (such as those used by Microsoft Office) can be reliably (albeit at a considerable performance cost compared to a proprietary extraction mechanism, but with significantly better results than in the case of simple extraction of printable text strings) converted into XML (and from there into plain text) using the OpenOffice import filters. Similar issues also exist with the popular Adobe PDF format; however, here elements of the GNU XPdf project can be used to extract plain text and encodings from within the extractor service. Regardless of the preceding steps, output of the text data extractor is a normalized ISO 10646 plaintext data stream that does not contain metadata.

### B. Image and Audio Data

Depending on the working environment (e.g. of analysts), significant amounts of data viewed may not be readily available in textual form but may consist of imagery, graphs, and audio data. While the concerns outlined in section II apply not only to textual data but also to multimedia data, automated analysis for content and particularly advanced feature and even concept extraction is clearly beyond what can be accomplished either within acceptable time or at all.

At the same time, information based on file system location is clearly insufficient since this neither provides sufficient granularity for distinction e.g. of multiple elements depicted in a single photograph and is subject to change when files are moved or copied. Annotation within file formats is also not sufficiently reliable since some file formats do not permit such annotation or sufficient amounts of it and are also subject to (deliberate or inadvertent) removal in case of format conversions or other manipulations that occur routinely.

The problem can, however, be addressed by embedding digital watermarks [19], [20] within the multimedia data prior to dissemination to workstations discussed in this paper; this permits the format- and (to a certain extent, depending on algorithms and embedding strenghts used) manipulation-independent annotation of multimedia data, e.g. using a system similar to that discussed in [21], [20]. The extraction of the requisite information for analysis as described in section VI can then be performed on the plaintext data extracted from the digital watermarks.

As in the case of complex textual data, the large number of data formats and dialects for encoding those multimedia data types represents a significant required effort that can be mitigated at the cost of lowered performance by transforming such file formats to a common intermediate format prior to attempts at recovery of the watermark. Moreover, it should be noted that each media type (even for two-dimensional images, different techniques are typically used for color and bitonal images as well as for vector-based representations) requires rather different marking techniques to achieve both acceptable recovery rates and quality of the marked media [19], [20]. However, in case of a successful detection of a markable media type and subsequent recovery of a digital watermark, the output of the extractor is again a normalized ISO 10646 plaintext data stream that does not contain metadata (typically only a selected number of keywords or an index into a database containing additional information for the given file[6]).

## VI. Semantic Modeling

The determination of anomalous behavior in accessing documents touches upon several highly active research areas in computer science and applied mathematics, including computational linguistics, natural language processing, statistical analysis, and formal concept analysis. For the purposes of this paper, it is assumed that the domain over which analysis is to occur is that of natural language (regardless of media types discussed in section V). Moreover, it is assumed that anomalies can be defined in terms of accessing documents containing concepts and keywords that are beyond the scope of a given tasking on the part of a user. This informal definition of an anomaly requires the distinction of several behavioral types:

- Document retrieval and processing within a given tasking can be characterized by a restriction to finite set of concepts.
- Conceptual drift results in the inclusion of a limited number of related concepts into the tasking concept set over time. Such behavior must be monitored to avoid a knowledgeable individual inducing slow, deliberate drift, thereby thwarting anomaly detection.
- Abrupt changes in the constitution of the concept set with limited overlap that stabilize once the shift has occurred can be assumed to indicate a new tasking.

The detection of anomalies requires that pertinent concepts can be identified automatically; this must occur at several levels. A prerequisite step is the creation of a concept dictionary along with thesauri for synonyms and related terms. While an initial dictionary, particularly of task-related terms must be built up manually, a number of techniques exist that permit automatic extension and derivation of such databases [23], [24], [25], [26], [27], [28], [29]; depending on the types of document an individual (e.g. intelligence analyst) is working on, this can also be extended to multilingual corpora [30], [31].

Based on such a concept groupings, a second problem that needs to be automated as far as possible is the separation of conceptual clusters through text categorization [32], [33]. A number of approaches has been proposed for this technique, including linear classifiers [34], context-sensitive learing mechanisms [35], Bayesian techniques [36], and decision trees [37], although typically a combination

[6]See [22] for a mechanism for such index annotation.

of techniques and algorithms is used, typically boosting-based classifier committees, support-vector machines and regression methods. Such classifiers exhibit adequate performance even for very large category sets [38]. In particular, boosted Bayesian networks have been successfully used on large corpora of documents (approx. $10^5$) and categories (approx. $10^4$) over extended periods [39], [40], [41], [42]; by combining multiple properties such as term properties, relations over terms and documents, and document properties (e.g. location in the file system, metadata attributes) and boosting multiple weak hypotheses, separation can be obtained with limited term occurrences [33].

Although the resulting data set is still of considerable dimensionality, a reduction of several orders of magnitude can be achieved by categorization as discussed above. The resulting data set can now be subdivided manually (e.g. using techniques from formal concept analysis [43], [44]) to achieve further dimensionality reduction; alternatively, analytical techniques suitable for such high dimensionality systems can be employed. In any case, one additional dimension (time) must be added to the data set.

Given the above processes, the behavioral anomalies described earlier can now be identified using statistical analysis techniques also commonly used in intrusion detection. A technique particularly suited for such analysis is multidimensional scaling (MDS) [45], [46], [47] followed by identifying centers of gravity for identified clusters and outliers from these clusters [48], [49], [50]; this can occur either automatically (based on fixed scaled thresholds) or in preparation for visual inspection (although in this case the mapping of a high dimensional space onto a two- or three-dimensional plot does not necessarily preserve structural properties such as linear separability of categories).

For each two documents of the observation set (obtained by restricting the data set to a given entity (individual) and a duration) $i, j$, a proximity metric $p_{ij}$ is defined such that $p_{ij}$ is smaller if the similarity between $i$ and $j$ is larger.

A configuration $X$ is constituted by $n$ points in an $m$-dimensional space and can be considered a $n \times n$ matrix of the coordinates of the $n$ points along $m$ axes of a cartesian coordinate system. The distance of points $i$ and $j$ in $X$, $d_{ij}$ can now be computed as

$$d_{ij} = (\sum_{a=1}^{m} \mid x_{ia} - x_{ja} \mid^m)^{\frac{1}{m}}$$

where $x_{kl}$ is the coordinate of point $k$ along the axis $l$ of the coordinate system. In the simplest (metric) case, the identity mapping is used to map the proximity measure ($f(p_{ij}) = p_{ij} = d_{ij}$) by way of a Minkowski distance; this, however, is justified only if the dissimilarity measure can be embedded in a metric space $(K, \delta)$ where $K$ is a set of points with $x, y, z \in K$ and $\delta(x, y)$ is a function $\delta : K \times K \rightarrow \mathbb{N}_0$ such that

$$\delta(x, y) = 0 \Leftrightarrow x = y \qquad \text{minimality} \qquad (1)$$

$$\delta(x, y) = \delta(y, x) \qquad \text{symmetry} \qquad (2)$$

$$\delta(x, z) \leq \delta(y, x) + \delta(y, z) \quad \text{triangle inequality} \qquad (3)$$

Nonmetric MDS [51], [45] employs arbitrary functions $f$ and merely assumes a monotonic relation between orderings of similarities and rank order of metric distances in a metric space; for the purposes of this discussion, however, metric MDS suffices. It should be noted, however, that these are merely basic examples of techniques that can be applied to the problem of identifying clusters and anomalies within the reduced semantic data space and may not necessarily provide optimum results.

## VII. Experiments

Since all analytical processes can be deferred to near-line status (and, apart from storage and network bandwidth considerations even need not be constrained to a single workstation but can be partitioned naturally for processing in grid-based systems), the main contributing factor to the usability of the Molehunt system is the degradation of service introduced by the interception processes. The impact of network interception was originally discussed in [17]; however, the interception mechanisms described in this paper do not introduce the caching latency discussed in [17].

For file systems, this requires consideration of two interception behaviors. First, immediate metadata access and copying of pages read from file systems. Here, worst-case unoptimized experiments performed on a low-end system (Intel Pentium II, 333 MHz, 256 MB RAM) running Microsoft Windows 2000 indicate that observed behavior is not affected; this is also supported by measurements.

The results of one such experiment are shown in figure 2; for this experiment, five files of 1 MB each were accessed twice and read into memory by an application program to obtain average read times (total elapsed as well as kernel and user mode times are shown). While initial access time was increased, average access times do not differ significantly even on markedly obsolete hardware; in some cases fluctuations in other operating system background activity dominates the additional overhead introduced.

The second type of behavior, backfilling, is significantly harder to quantify since its impact depends heavily on the amount of memory (cache) pages available, the maximum I/O load experienced by a system until cache contents must be written out, and the amount of cache churning that occurs. Here, worst-case unoptimized experiments indicate that for mixed media document viewing and browsing on the low-end PC system, observed behavior is affected minimally; this is mainly due to a higher incidence of interrupt requests issued by the disk subsystem.

## VIII. Related Work

The author is not aware of similar systems in the application domain being discussed in the open literature; in the

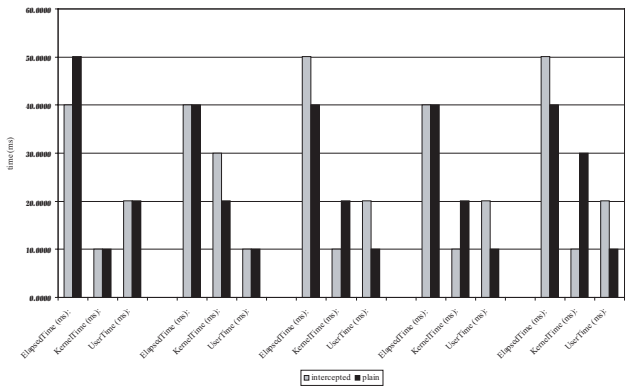


Fig. 2. Average access time for 1 MB files completely read (intercepted: light gray).

context of messaging and mail guards, such classification mechanisms have been described by Monteith *et al.* [52], [53]. Simple keyword-based filtering and analysis on message traffic has e.g. been used extensively in guard architectures [54], [55], [56] while the use of intelligent systems for knowledge and structural extraction has been explored for an intelligence workstation system (Intelligence Analyst Associate) and the Cyc system by Lenat *et al.* [57], [58].

## IX. Conclusion and Outlook

This paper has described a system for monitoring and analysis of document access and information retrieval primarily at the individual file level and detecting abnormal behavior in situations where elevated protection levels for enforcing high granularity are not desirable or feasible. The mechanisms described do require a significant amount of manual intervention and analysis and are therefore presumably limited to environments where privacy concerns are secondary, such as in counterintelligence [6]. Similarly, the semantic clustering of terms and phrases is most effective when applied to a specific problem domain where the creation, maintenance, and pruning of concept dictionary and thesauri can be accomplished with reasonable effort.

A number of research strands should be further explored. These include work on the inclusion of category-theoretic lattices to guide the categorization and clustering with domain-specific knowledge as well as the exploration of alternative techniques to establish cluster ensembles and to detect cluster outliers (anomalies). A number of techniques commonly applied in intrusion detection systems and also in data mining for the analysis of multidimensional data sets can be analyzed. In addition, visualization techniques for analysis, particularly the ability to recall documents based on a dimensionally reduced represenation such as that provided by an MDS application may assist analysts in assessing the permissibility of certain types of behavior.

## References


[1] United States Department of Defense, Washington D.C., USA, *Department of Defense Directive 5220.22-M: National Industrial Security Program Operating Manual (NISPOM)*, Dec. 2002.

[2] W. H. Webster, "A Review of FBI Security Programs," tech. rep., U.S. Department of Justice Commission for Review of FBI Security Programs, Washington D.C., USA, Mar. 2002.

[3] G. A. Fine, "A Review of the FBI's Performance in Deterring, Detecting, and Investigating the Espionage Activities of Robert Philip Hanssen," tech. rep., U.S. Department of Justice Office of the Inspector General, Washington D.C., USA, Aug. 2003. Unclassified executive summary; full report is classified at Top Secret/Codeword level.

[4] D. Wise, *Spy: The Inside Story of How the FBI's Robert Hanssen Betrayed America*. New York, NY, USA: Random House, 2002.

[5] J. B. Bruce, "The Consequences of Permissive Neglect: Laws and Leaks of Classified Intelligence," *Studies in Intelligence*, vol. 47, no. 1, pp. 39–49, 2003.

[6] G. F. Jelen, "The Nature of OPSEC," *The OPSEC Journal*, vol. 1, no. 1, pp. 1–12, 1993.

[7] D. Solomon, *Inside Windows NT*. Bellevue, WA, USA: Microsoft Press, 2nd ed., 1998.

[8] D. Solomon and M. Russinovich, *Inside Windows 2000*. Bellevue, WA, USA: Microsoft Press, 3rd ed., 2000.

[9] R. Goldenberg and S. Saravanan, *Open VMS AXP Internals and Data Structures: Version 1.5*. Maynard, MA, USA: Digital Press, 1994.

[10] B. Goodheart and J. Cox, *The Magic Garden Explained: The Internals of Unix System V Release 4*. Englewood Cliffs, NJ, USA: Prentice Hall, 1994.

[11] S. Wolthusen, "Security Policy Enforcement at the File System Level in the Windows NT Operating System Family," in *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)*, (New Orleans, LA, USA), pp. 55–63, IEEE Press, Dec. 2001.

[12] R. Nagar, *Windows NT File System Internals: A Developer's Guide*. Sebastopol, CA, USA: O'Reilly & Associates, 1997.

[13] M. Russinovich and B. Cogswell, "Examining the Windows NT Filesystem," *Dr. Dobb's Journal of Software Tools*, vol. 22, pp. 42–50, Feb. 1997.

[14] H. Custer, *Inside the Microsoft Windows NT File System*. Redmond, WA, USA: Microsoft Press, 1994.

[15] P. Dabak, S. Phadke, and M. Borate, *Undocumented Windows NT*. Foster City, CA, USA: M&T Books, 1999.

[16] G. Nebbett, *Windows NT/2000 Native API Reference*. Indianapolis, IN, USA: Macmillan Technical Publishing, 2000.

[17] E. Rademer and S. Wolthusen, "Transparent Access To Encrypted Data Using Operating System Network Stack Extensions," in *Communications and Multimedia Security Issues of the New Century: Proceedings of the IFIP TC6/TC11 Fifth Joint Working Conference on Communications and Multimedia Security (CMS'01)* (R. Steinmetz, J. Dittman, and M. Steinebach, eds.), (Darmstadt, Germany), pp. 213–226, IFIP, Kluwer Academic Publishers, May 2001.

[18] S. Wolthusen, "Tempering Network Stacks," in *Proceedings of the NATO RTO Symposium on Adaptive Defense in Unclassified Networks*, (Toulouse, France), NATO Research and Technology Organization, Apr. 2004.

[19] I. J. Cox, M. L. Miller, and J. A. Bloom, *Digital Watermarking*. The Morgan Kaufmann Series in Multimedia Information and Systems, San Francisco, CA, USA: Morgan Kaufmann Publishers, 2002.

[20] M. Arnold, M. Schmucker, and S. D. Wolthusen, *Techniques and Applications of Digital Watermarking and Content Protection*. The Artech House Computer Security Series, Norwood, MA, USA: Artech House, 2003.

[21] C. Busch and S. Wolthusen, "Tracing Data Diffusion in Industrial Research with Robust Watermarking," in *Proceedings of the 2001 Fourth Workshop on Multimedia Signal Processing (MMSP'01)* (J.-L. Dugelay and K. Rose, eds.), (Cannes, France), pp. 207–212, IEEE Press, Oct. 2001.

[22] C. Busch and S. Wolthusen, "Sensitivity Labels and Invisible Identification Markings in Human-Readable Output," in *Proceedings of Electronic Imaging 2002*, (San Jose, CA, USA), pp. 149–157, The International Society for Optical Engineering (SPIE), Jan. 2002.

[23] C. J. Crouch, "An Approach to the Automatic Construction of Global Thesauri," *Information Processing and Management*, vol. 26, no. 5, pp. 629–640, 1990.

[24] M. W. Berry, ed., *Survey of Text Mining: Clustering, Classification, and Retrieval*. Heidelberg, Germany: Springer-Verlag, 2003.

[25] P. P. Senellart and V. D. Blondel, "Automatic Discovery of Similar Words," in Berry [24].

[26] C. Fellbaum, ed., *WordNet: An Electronic Lexical Database*. Cambridge, MA, USA: MIT Press, 1998.

[27] M. Lesk, "Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Code from an Ice Cream Cone," in *Proceedings of the 5th Annual International Conference on Systems Documentation*, (Toronto, ON, Canada), pp. 24–26, Springer-Verlag, June 1986.

[28] S. Banerjee and T. Pedersen, "An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet," in *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLING-02)*, vol. 2276 of *Lecture Notes in Computer Science*, (Mexico City, Mexico), pp. 805–810, Springer-Verlag, Feb. 2002.

[29] S. Banerjee and T. Pedersen, "Extended Gloss Overlaps as a Measure of Semantic Relatedness," in *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-2003)*, (Acapulco, Mexico), pp. 805–810, Morgan Kaufmann, Aug. 2003.

[30] M. T. Pazienza, ed., *Information Extraction: Towards Scalable, Adaptable Systems*, vol. 1714 of *Lecture Notes in Artificial Intelligence*. Heidelberg, Germany: Springer Verlag, 1999.

[31] H. Somers, "Knowledge Extraction from Bilingual Corpora," in Pazienza [30].

[32] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. Cambridge, MA, USA: MIT Press, 1999.

[33] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, vol. 54, no. 1, pp. 1–47, 2002.

[34] D. D. Lewis, R. E. Schapire, J. P. Callan, and R. Papka, "Training Algorithms for Linear Text Classifiers," in *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (Zurich, Switzerland), pp. 298–306, ACM Press, Aug. 1996.

[35] W. W. Cohen and Y. Singer, "Context-Sensitive Learning Methods for Text Categorization," *ACM Transactions on Information Systems*, vol. 17, no. 2, pp. 141–173, 1999.

[36] T. T. Makoto Iwayama, "Hierarchical Bayesian Clustering for Automatic Text Classification," in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-1995)*, (Montreál, QC, Canada), pp. 1322–1327, Morgan Kaufmann, Aug. 1995.

[37] C. Apté, F. Damerau, and S. M. Weiss, "Automated Learning of Decision Rules for Text Categorization," *ACM Transactions on Information Systems*, vol. 12, no. 3, pp. 233–251, 1994.

[38] Y. Yang, J. Zhang, and B. Kisiel, "Text Categorization: A Scalability Analysis of Classifiers in Text Categorization," in *Proceedings of the 26th Annual International ACM SIGIR conference on Research and Development in Information Retrieval*, (Toronto, ON, Canada), pp. 96–103, ACM Press, July 2003.

[39] G. Knorz, "A Decision Theory Approach to Optimal Automatic Indexing," in *Proceedings of the 5th Annual ACM Conference on Research and Development in Information Retrieval*, (Berlin, Germany), pp. 174–193, ACM Press, May 1982.

[40] K. Tzeras and S. Hartmann, "Automatic Indexing Based on Bayesian Inference Networks," in *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (Pittsburgh, PA, USA), pp. 22–35, ACM Press, June 1993.

[41] N. Fuhr, "A Probabilistic Model of Dictionary Based Automatic Indexing," in *Proceedings of RIAO-85, First International Conference "Recherche d'Information Assistee par Ordinateur"*, (Grenoble, France), pp. 207–216, Mar. 1985.

[42] N. Fuhr, S. Hartmann, G. Lustig, M. Schwantner, K. Tzeras, and G. Knorz, "AIR/X a Rule-Based Multistage Indexing System for Large Subject Fields," in *Proceedings of RIAO-91, Third International Conference "Recherche d'Information Assistee par Ordinateur"*, (Barcelona, Spain), pp. 606–623, Apr. 1991.

[43] B. Ganter and R. Wille, *Formal Concept Analysis – Mathematical Foundations*. Heidelberg, Germany: Springer Verlag, 1998. Originally released in German as "Formale Begriffsanalyse – Mathematische Grundlagen".

[44] U. Priss, "Linguistic Applications of Formal Concept Analysis," in *Proceedings of the First International Conference on Formal Concept Analysis (ICFCA 2003)*, Lecture Notes in Artificial Intelligence, (Darmstadt, Germany), Springer-Verlag, Mar. 2003. To appear.

[45] J. Kruskal and M. Wish, *Multidimensional Scaling*, vol. 07-011 of *Sage University Paper Series on Qualitative Applications in the Social Sciences*. London, UK: Sage Publications, 1978.

[46] S. Kotz, N. L. Johnson, and C. B. Read, eds., *Encyclopedia of Statistical Sciences*, vol. 5. New York, NY, USA: John Wiley & Sons, Inc., 1985.

[47] F. W. Young, "Multidimensional Scaling," in Kotz *et al.* [46].

[48] J. van Ryzin, ed., *Classification and Clustering*. New York, NY, USA: Academic Press, 1977.

[49] J. B. Kruskal, "The Relationship between Multidimensional Scaling and Clustering," in van Ryzin [48].

[50] A. Strehl, *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. PhD thesis, University of Texas at Austin, Austin, TX, USA, 2002.

[51] J. B. Kruskal, "Multidimensional Scaling by Optimiziing Goodness of Fit to a Nonmetric Hypothesis," *Psychometrika*, vol. 29, pp. 1–27, 1964.

[52] E. Monteith, "Genoa TIE, Advanced Boundary Controller Experiment," in *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)*, (New Orleans, LA, USA), pp. 74–82, IEEE Press, Dec. 2001.

[53] J. Epstein, "Architecture and Concepts of the ARGuE Guard," in *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)*, (Phoenix, AZ, USA), pp. 45–54, IEEE Press, Dec. 1999.

[54] C. L. Heitmeyer and C. E. Landwehr, "Designing Secure Message Systems: The Military Message Systems (MMS) Project," in *Proceedings of the IFIP TC6.5 Working Conference on Computer-Based Message Services*, (Nottingham, UK), pp. 245–255, IFIP, Kluwer Academic Publishers, May 1984.

[55] R. E. Smith, "Constructing a High Assurance Mail Guard," in *Proceedings of the 17th National Computer Security Conference*, (San Diego, CA, USA), pp. 247–253, Oct. 1994.

[56] R. E. Smith, "Cost Profile of a Highly Assured, Secure Generating System," *ACM Transactions on Information and System Security*, vol. 4, pp. 72–101, Feb. 2001.

[57] D. B. Lenat and R. V. Guha, eds., *Building Large Knowledge Based Systems*. Reading, MA, USA: Addison-Wesley, 1990.

[58] J. Neal, B. Rode, C. Crowner, and D. Gunning, "Intelligence Analyst Associate (IAA): CYC Knowledge Extraction," tech. rep., Veridian Engineering / General Dynamics Advanced Information Systems, Buffalo, NY, USA, Jan. 2003.