Combining a Baiting and a User Search Profiling Techniques for Masquerade Detection

Abstract. Masquerade attacks are characterized by an adversary stealing a legitimate user's credentials and using them to impersonate the victim and perform malicious activities, such as stealing information. Prior work on masquerade attack detection has focused on profiling legitimate user behavior and detecting abnormal behavior indicative of a masquerade attack. Like any anomaly-detection based techniques, detecting masquerade attacks by profiling user behavior suffers from a significant number of false positives. We extend prior work and provide a novel integrated detection approach in this paper. We combine a user behavior profiling technique with a baiting technique in order to more accurately detect masquerade activity. We show that using this integrated approach reduces the false positives by 36% when compared to user behavior profiling alone, while achieving almost perfect detection results. We also show how this combined detection approach serves as a mechanism for hardening the masquerade attack detector against mimicry attacks.

1 Introduction

Masquerade attacks, in which attackers impersonate the legitimate user of a computer system, are increasingly growing in number and gravity. They were classified second in the top five list of most frequent electronic crimes performed by outsiders against organizations according to the 2010 e-crime watch survey [8]. 35% of the surveyed executives and law enforcement officials indicated that they experienced such attacks against their organizations. The best known example of masquerade attacks is identity (ID) theft in financial transaction systems. Fifteen million Americans were ID theft victims in 2006 alone according to Gartner, with average losses of more than $3,200 for each [27].

Masqueraders can steal a legitimate user's credentials by sniffing passwords, using password crackers, or installing a keylogger, etc. Whatever the access method to the victim's computer is, access control mechanisms cannot be used to detect the masquerader. Detecting masquerade attacks is therefore very difficult. Most proposed approaches rely on profiling legitimate user behavior by auditing a variety of sources, such as command line calls issued by users, system calls, application events, database and file accesses, and the organization policy management rules, and compliance logs. The algorithms used for modeling user behavior use statistical features, such as the sequence of user commands or co-occurrence of multiple events combined through logical operators. The anomaly detectors built using these algorithms are used then to flag deviations from normal user behavior. The assumption is that behavior that is widely inconsistent/different from the user's historical behavior could be attributed to someone else, a masquerader.

Anomaly detectors, however, suffer from low accuracy, and particularly from high false positive (FP) rates. One way to overcome this shortcoming is by combining

several base classifiers into one ensemble classifier. Each classifier uses a different modeling algorithm to profile user behavior. Base models can be aggregated by learning from labeled data or by achieving consensus among the individual models. The ensemble methods output collectively one classification label which reflects the meta-learning from these models or the consensus amongst them.

Techniques for creating ensemble classifiers using multiple diverse classifiers have been studied extensively [10, 14, 12, 13]. The objective of using such ensemble methods is to improve robustness and classification accuracy over single-model methods. Improvement in classification accuracy, however, can only be achieved if the base models are mutually independent. This conditional independence assumption may not always hold true though.

In the absence of the independence condition, how effective are these ensemble anomaly detectors, *i.e.* how effective is *model diversity*? Tan and Maxion studied the effects of using diverse anomaly detection algorithms on detector performance [31]. They investigated how various sequence-based anomaly detectors dealt with a specific anomaly, namely a 'foreign' sequence, *i.e.* a sequence that has never been seen during the training phase of the detection algorithm. Their results showed that limited performance/anomaly coverage gains can be achieved by combining various anomaly detection algorithms. The anomaly coverage gains are mostly seen at the edges of the anomaly space. This indicates that such gains are highly dependent on the characteristics of the detected anomaly and on the parameter settings of the anomaly detector. Furthermore, these limited gains may not justify the additional classifier training and deployment cost and performance overhead that are likely introduced through the combination of different classifiers.

Moreover, if the anomaly space of various classifiers is mostly overlapping, the ensemble method does not offer any additional protection mechanism against mimicry attacks. Note that anomaly detectors are subject to mimicry attacks, where the attacker tries to mimic the legitimate user's behavior. In such case, the attacker's activities will not be detected as abnormal, and therefore they manage to escape detection. If the different classifiers have highly overlapping anomaly spaces, then when evading detection by one classifier by mimicking normal user behavior, an attacker is likely to escape detection by the other classifiers. Combining different classifiers in this case does not constitute a defense mechanism against mimicry attacks.

To overcome the limitations of model diversity, we propose diversifying the detection techniques. We combine two different detection techniques for the purpose of detecting masquerade attacks. The first is an anomaly detection technique based on profiling user search behavior. The second is a baiting technique where access to decoy documents is monitored. Decoys files are strategically placed by the legitimate user in their own file system. The user is not supposed to touch these files after installing them on their system. Any access to these decoy document is then considered as indicative of masquerade activity and therefore triggers an alert.

The two detection techniques are **orthogonal**. We conjecture that, if a masquerade attack takes place, it can be manifested in both data streams that are monitored and modeled by the individual detection techniques, namely user search behavior on the victim's system, and touches to decoy documents, even though the two data

streams remain relatively independent in the absence of masquerade attacks. Based on this conjecture, we show that combining the two techniques can be used to improve the accuracy results of a masquerade attack detector. Prior work using the search behavior profiling approach alone achieved a 100% detection rate with a 1.1% (FP) rate [5]. Our objective is to substantially reduce this FP rate, without significanlty affecting the true positive (TP) or detection rate. We also show that the combination of the two techniques can be used as a defense mechanism against mimicry attacks targeted at any user behavior anomaly detector.

The contributions of this work include:

- An **integrated approach for masquerade attack detection** which combines user behavior profiling with a baiting approach that makes use of highly-crafted and well-placed decoy documents to bait attackers. The approach improves detection accuracy over prior techniques and is less vulnerable to mimicry attacks.
- A host-sensor that implements this integrated detection approach and collects potential evidence that could be used to identify the attacker.

The rest of this paper is organized as follows. In Section 2, we briefly review prior related work. Section 3 expands on the objective and the approach taken in this work. Section 4 presents the experiments conducted to evaluate how the combination of the user behavior profiling and the baiting techniques improve the accuracy of our masquerade attack detector. Section 5 demonstrates how the combined detection approach serves as a defense mechanism against mimicry attacks. Finally, section 6 concludes the paper by summarizing the contributions of this paper and presents directions for our future work.

2 Related Work

We first start by giving a brief overview of prior work on user behavior profiling for masquerade attack detection. Then, we discuss some prior work that used baiting techniques for intrusion detection. Finally, we present approaches that used diverse or integrated techniques for masquerade detection.

2.1 User Behavior Profiling

Most of the prior user behavior profiling work focused on auditing and modeling sequences of user commands including work on enriching command sequences with information about command arguments [28, 21, 32, 22, 20, 11, 26, 25, 35]. A thorough review of these machine learning techniques can be found in this survey [2]. The detection rates of these anomaly detection techniques ranged between 75.8% and 26.8%, with FP rates ranging between 1% and 7%. These results are obviously far from satisfactory.

Maloof and Stephens also applied a user behavior profiling technique to detect malicious insider activities which violated 'Need-to-Know' policy [19]. In order to identify bad insider behavior, they defined the malicious user scenarios and had to combine results from 76 different sensors through a Bayesian net. Although the few

attack scenarios tested were detected, there was no real evaluation of the FP rate associated with the overall classifier.

Finally, Ben-Salem and Stolfo proposed a search-behavior profiling approach for detecting masquerade attacks [5]. The authors focused on modeling user search behavior to reveal an attacker's malicious intent. They hypothesized and showed that a masquerader would engage in search activities different from those of the legitimate user in terms of their volume and frequency. They showed substantial accuracy improvement reaching a 100% detection rate and a 1.1% FP rate on a home-gathered Windows dataset with simulated masquerader attacks [1].

2.2 Honeypots and Honeytokens

Honeypots are information systems used to deceive adversaries and trick them into thinking that they are dealing with real and authentic assets. Honeypots have been widely deployed in De-Militarized Zones (DMZ) to trap attempts by external attackers to penetrate an organization's network. Spitzner proposed the use of honeypots within the network of an enterprise in order to detect insider attacks [29]. He introduced the concept of 'honeytokens' such as user credentials that can be embedded in database tables, or decoy files placed in a file system. Decoy files, or 'honeyfiles', were further developed by Yuill et al. [34, 33]. The authors created a system that allows users to select files from the user space on a network file server, and change them into decoy files. Illegitimate access to the honeyfiles can then be monitored by consulting a record that associates the honeyfile with the legitimate userid.

Bowen et al. proposed an automated system for generating decoy documents [7, 6]. The system generated files from different templates with various themes, such as a health-related information theme, a financial accounts theme, or a tax returns theme.

The use of honeyfiles may not be very effective if used alone, as the attacker may never access the decoy file. Several measures can be taken to maximize the likelihood that the adversary stumbles upon the honeyfile, such as increasing the conspicuousness and enticingness of the file [7, 3]. However, there is a risk that an intrusion does not get detected. In this paper, we propose to supplement monitoring access to decoy files on a host with profiling user behavior in order to get more coverage for suspicious activity that could be indicative of a masquerade attack.

2.3 Diversity in Intrusion Detection Systems

Diversity is an approach that has been widely used in fault-tolerant and self-healing systems for developing robust systems. The first attempt to apply the concept of *diversity* to computer security was made by Littlewood and Strigini [18]. Motivated by the application of models of diversity in system and process design and by the work on formal probability modeling of reliability and safety [24], the authors studied the roles and limits of redundancy and diversity in intrusion detection systems. They suggested that modeling diversity could be utilized to build IDS systems that provide more coverage to intrusion attacks. They argued for a formal mathematical approach to estimating the effectiveness of both approaches and for a metric

for measuring the independence of various intrusion detection systems (IDSs) by category of attack rather than by some average mixture of attacks.

Gashi et al. [15] studied the actual gains in detection rates that can be obtained through the use of *diverse* or *different* off-the-shelf anti-virus engines. They showed that when using only two anti-virus engines, almost a third of the engine pairs perform better than the best individual engine.

Tan and Maxion studied the anomaly space of several different sequence-based anomaly detectors when presented with a 'foreign sequence', *i.e.* a never-before-sequence of events, as an anomaly [31]. They showed the anomaly spaces of these anomaly detectors are highly overlapping, which limits or eliminates any potential detection accuracy gains that could be achieved by combining several anomaly detectors into one classifier.

A system composed of honeypots and network-level sensors for traffic profiling was proposed by Maybury et al. [23]. The sensors monitored insider activities such as network scanning and file downloads. Pre-specified models of insiders and pre-attack indicators were used to infer the malicious intent of an insider. The authors however did not report any test and evaluations. In this paper we integrate host-level user monitoring with honeytokens, as opposed to network-level monitoring that Maybury et al. proposed, and we provide a thorough evaluation of the integrated approach.

3 Motivation and Approach

In systems, the concept of diversity is applied to system design, process, as well as argument diversity. Diversity can be applied to anomaly detectors along different dimensions listed below:

1. Diversity in the design of IDSs, thus providing reliability when sensors are subject to the same software/system attack.
2. Diversity in modeling algorithms, as some algorithms are more suitable for certain user profiles and behaviors than others. For example, support vector machines may not make the best classifier in the case of a user whose behavior is closest to the 'average' user behavior [4].
3. Diversity of features used by one modeling algorithm in order to accurately model the unique and distinct user behavior.
4. Diversity of data streams and events used for modeling by the anomaly detector.

'Algorithmic diversity' does not necessarily improve detection accuracy due to the highly overlapping anomaly spaces [31]. However, combining diverse and orthogonal detection technique may provide such improvements. We chose to combine a user search behavior profiling technique with a baiting technique. The choice is not arbitrary. Below we discuss some of the factors that guided our selection of detection techniques:

User behavior is not readily available: User behavior is not readily available for stealing and use (assuming the historical information profiled is kept secret).

Search volume is correlated with masquerade activity: The legitimate user of the system is familiar with the files on the system and where they are located.

Any search for specific files is likely to be targeted and limited. A masquerader, however, who gets access to the victim's system illegitimately, is not familiar with that environment. The attacker is therefore likely to engage in a wide information gathering exercise before launching any attack. Their search is likely to be widespread and untargeted. Prior work showed that user search behavior profiling can detect masquerade attacks reliably with low FP rates [5].

Decoy files can bait masqueraders: We use decoy files that contain "bait information" such as online banking logins, social security numbers, and web-based email account credentials. Legitimate users can place different types of these decoy documents, such as tax return forms, medical records, credit card statements, e-bay receipts, etc., on their personal file systems. Once in place, the legitimate users of the system are supposed to avoid accessing these decoy files. We monitor access to these decoy files. Any decoy file access is taken as an anomaly, and is considered therefore, indicative of a masquerader trying to access highly enticing information. Not only would a masquerader not know the file system, they would also not know the detailed contents of that file system including the well-placed traps and decoys.

Besides detecting masqueraders, placing monitorable decoy files on the system has a deterrence effect, which may not be easily measurable, but which definitely plays a role in preventing masquerade activity by risk-averse attackers.

Two techniques with complementary strengths and weaknesses: Profiling user behavior is an anomaly detection technique. Anomaly detection has the potential to detect new and unknown attacks, and when profiling user behavior, any abnormal behavior suggestive of masquerade activities. However, it may produce high FP rates, particularly if the user model does not generalize well because of data over-fitting or lack of model training data. On the other hand, trap-based techniques are known for their very low FP rates. Combining a technique that provides a wide attack coverage (*i.e.* a low miss rate), with a technique that has a very low FP rate may improve overall detection accuracy.

Defense mechanism against mimicry attacks: Anomaly detection-based techniques are vulnerable to mimicry attacks. An adversary may know how a user behaves and execute a 'mimicry attack'. However, they are unlikely to know what the victim knows. If the victim baits the system with well-placed decoys, the latter may trap the adversary. Even a sophisticated adversary who mimics the victim or target user may still get trapped, as they do not know where the decoys were placed.

Therefore, combining diverse and orthogonal techniques opposed to diverse anomaly detection algorithms auditing the same data, could serve as a defense mechanism against mimicry attacks. If the anomaly detector is subject to such an attack, then the decoy file monitoring sensor is likely to catch the masquerade activity, something that may not hold true if two anomaly detectors are used where the non-anomaly space is highly overlapping.

Orthogonal techniques: Detecting an attack using two orthogonal techniques where two independent data streams are monitored provides stronger evidence of the attack. So the correlation of search behavior anomaly detection with trap-based decoy files should provide stronger evidence of malfeasance, and therefore improve the detector's accuracy.

We hypothesize that detecting abnormal search operations performed prior to an unsuspecting user opening a decoy file will corroborate the suspicion that the user is indeed impersonating another victim user. Furthermore, an accidental opening of a decoy file by a legitimate user might be recognized as an accident if the search behavior is not deemed abnormal. In other words, detecting abnormal search and decoy traps together may make a very effective masquerade detection system.

3.1 Threat Model

We assume that the adversary knows that their activity on the victim's system is monitored. We also assume that the attacker does not know that system is baited with decoy documents. In all cases, we assume that the attacker can access the information assets that need to be protected from inside the system via trusted access or system compromise. Therefore, we do not focus on monitoring access attempts to the system. The protected assets could range from Personally Identifiable Information (PII) (e.g. customer records and employee personal data), to user credentials to Intellectual Property (IP), and other sensitive data (e.g. financial records) stored on the target system. We do not address the case of traitors that have full administrator privileges and full knowledge of the system in multi-user systems such as file servers.

3.2 Detection Approach

We developed a sensor to detect data theft attempts performed by masqueraders on single-user systems. The sensor can also be modified to detect "need-to-know" policy violations perpetrated by traitors on multi-user systems, such as file systems. We refer to this sensor as the RUU (Are You You?) sensor. The sensor is composed of two sub-sensors as displayed in Figure 1. The first sensor is a s User Search Behavior (USB) sensor. As its name indicates, this sensor profiles user search behavior, and detects any abnormal search activity. The second sensor is a Decoy Documents Access (DDA) sensor, which monitors any access to the decoy documents embedded in the file system. It also serves as an oracle for the USB sensor.

The sensor also includes a monitoring infrastructure which ensures that the sensor does not get tampered with. However, explaining how we ensure the tamper-resistance of the sensor is out of the scope of this paper.

The integrated RUU sensor provides three mitigation strategies when it suspects malicious masquerade activity. These strategies can be selectively implemented depending on the confidence level of the sensor that malicious activity is taking place:

1. Sending an alert message to a remote server
2. Displaying a set of challenge-response questions that the user must correctly respond to: The answers to these questions are given by the owner of the system during the installation of the sensor.
3. If a webcam is available, stealthily recording audio and taking a picture: The data will be kept on the system and could be used as evidence against the malefactor if a masquerade attack did indeed take place.

Here we describe how each component of the RUU sensor works, and how the USB and DDA sensors are integrated in order to detect masquerade attacks.

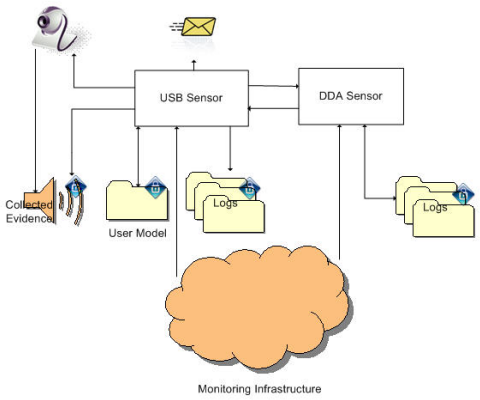


Fig. 1. Architecture of the RUU Masquerade Attack Sensor

Profiling Search Behavior The USB sensor detects abnormal user search behavior after profiling user actions and forming a baseline of *search* behavior utilizing anomaly detection techniques. Then it monitors for abnormal search behaviors that exhibit large deviations from the baseline. Such deviations signal a potential masquerade attack.

The sensor builds a one-Class Support Vector Machine (ocSVM) model that models the user's search behavior. Vectors with three search-related features are extracted for each two minute period of user activity. The three search behavior-related features are:

1. Number of **automated** search-related actions: Specific sections of the Windows registry, specific Dynamic Link Libraries (DLLs), access to specific index files, and specific programs, particularly desktop search tools, are correlated with system searching. The total number of these search-related events are modeled per 2-minute epoch.
2. Number of file touches: Any file fetch, read, write, or copy action results into loading the file into memory. The number of times files are touched and loaded into memory by any process within each 2-minute epoch is used as a feature.
3. Percentage of file system navigation user actions: Manually exploring a file system and navigating through file system's directories represents a form of user search, albeit different from the automated search that relies on desktop search tools. All **manual search** or file system navigation user activity occurring during the 2-minute epoch is monitored and extracted as a feature.

We identify two thresholds per user model which we use to classify the observed user search activity as normal, abnormal, or non-identifiable. The first threshold thr_1 is determined empirically, so that the miss rate or false negative rate is minimized. A second threshold thr_2 is also set to minimize the FP rate. During the detection phase, we continuously monitor user search activity, and extract a feature vector v every two minutes. We measure the deviation d between actual user behavior

and the historical user behavior as defined by the user model u. The distance d is compared to thr_1 and to thr_2 in order to determine whether there is enough evidence for masquerade activity. We explain this in more details in section 3.2 .

Monitoring Access to Decoy Documents We use decoy documents that carry a keyed-Hash Message Authentication Code (HMAC) [17] embedded in the header section of the document, and visible only if the document is opened using a hex editor. The HMAC is computed over a file's contents using a key unique to the user, and is hidden in the header section of the file. The decoy files can be downloaded by the legitimate user of the system from the Decoy Document Distributor (D^3) [6]. (D^3) is a distribution platform that offers several types of decoy documents such as tax return forms, medical records, credit card statements, e-bay receipts, etc..

The DDA sensor detects when decoy documents are being read, copied, or zipped. As soon as the decoy document is loaded into memory by any application or process, the sensor initiates a verification function, which checks whether the file is normal or a decoy by computing a HMAC based on all the contents of that file and comparing it to the one embedded within the document. If the two HMACs match, the document is deemed a decoy; otherwise, the document is deemed normal.

During the deployment of the decoy documents, the user can take certain actions to increase the conspicuousness and the enticingness of these decoys. Other actions can also be taken to reduce the interference of these decoys with the user's normal activities and to minimize false alerts by the sensor that are not related to masquerader activities. For recommendations on how to maximize the effectiveness of the decoy documents in detecting masquerader activity, we refer the reader to this study which evaluates different properties and characteristics of decoy files and how they can be utilized for highly effective masquerade detection [3].

Integrated Masquerade Detection Approach We use the DDA sensor as an oracle for the USB sensor. As explained in section 3.2, two detection thresholds are defined for each user search model thr_1 and thr_2. Recall that thr_1 is set, so as to minimize the miss rate or false negative rate. If the user behavior captured in feature vector v is similar enough with the user model u which captures the user's historical behavior, then the user behavior should be deemed normal. In other words, if the distance d between the v and user model u is smaller than thr_1, then no masquerader activity is suspected, and no alert gets generated. If, on the other hand, feature vector u exhibits a highly abnormal search, that is if $d > thr_2$, then an alert is generated. However, if $thr_1 < d \leq thr_2$, then the USB sensor checks whether any excessive access to decoy documents has been recorded by the DDA sensor. If so, then an alert is generated and the right mitigation strategy is initiated. Otherwise, the user search activity is not deemed suspicious enough. Figure 2 describes the overall decision process related to masquerade alert generation using the two sensors.

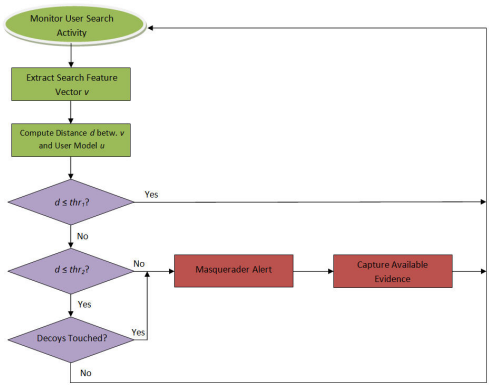


Fig. 2. Alert Generation Decision Process

4 Experimental Evaluation

4.1 Data Gathering

Normal User Data Collection In order to evaluate the integrated detection approach, we gathered both normal user data and simulated masquerader data, that could be used to build user search models and test for abnormal search and access to decoy documents. To do so, we conducted two user studies. Eighteen computer science students participated in the first user study. Nine of them placed 20 decoy documents, and nine others placed 30 decoy documents on their local file systems. When placing the decoy files, the students were encouraged to select file locations such that the conspicuousness of the decoys is increased, while the non-interference with user activities is minimized.

The participants in this user study also installed a host sensor on their personal machines, which audited their search activity, registry-based activity, process creation and destruction, window GUI access, DLL libraries activity and decoy document accesses. The data collected by the host sensor were automatically and periodically uploaded to a central server for analysis. The experiment lasted for about one week.

Masquerader Data collection We also conduced a 'capture-the flag' exercise where we simulated masquerade attacks. We recruited 40 students to pose as masqueraders trying to steal personal or confidential information from a coworker's desktop during 15 minutes. The masqueraders in these simulated attacks were motivated by stealing personal identifiable information and other data that could be used for financial gain. All participants were given the same scenario that described the attack type.

We set up a computer in our lab that served as the victim' computer in these simulated masquerade attacks. We installed various applications and programs typically used by computer science students. The objective is to make the victim's computer resemble as much as possible the computers from which we collected 'normal

user' data. We placed 80 files with personal or confidential information that could be used by an attacker for financial gain, and therefore could constitute a target for the adversary who wants to steal data, throughout the file system. We also placed 40 decoy files. The locations of these decoy files were not chosen arbitrarily. To make sure that our test environment reflects the real victim' environment as closely as possible, we surveyed the students who participated in out first user study. They reported where they placed their decoy files. on their file systems. We have selected the top 40 locations most identified by students, and used them to place the decoy documents in our user study.

We ran four experiments. We started with 40 decoy files in the first experiment and randomly selected 10 participants from our volunteers to act as masqueraders. In the second experiment, we reduced the number of decoy files to 30, all located in the top 30 mos popular locations selected by our normal user population. Similarly, the third and fourth experiments included 20 and 10 decoy files respectively.

4.2 Experimental Setup

We used 80% of the data to train one-class SVM models usng features as described in Section 3.2. We used the LibSVM tool kit to build the models [9]. We also developed a linear decoy access classifier for each user, which checks the history of accidental accesses of the user to the decoy documents on their file system. Based on this hostorical behavior, we select a threshold, beyond which access to decoy documents is considered excessive or suspicious, in other words indicative of masquerade acivity. These models are also developed for each individual user by using 80% of the decoy access data. We used the rest of the user data, as well as the simulated masquerader data for testing the user models.

4.3 Detection Accuracy

We ran two different experiments. In the first experiment we developed and tested one-class SVM models using the same search profiling approach presented by Ben-Salem and Stolfo [5]. In the second experiment we supplemented these models with a linear classifier based on decoy file accesses, as described in Section 3.2 and Figure 2.

Table 1 shows that using the combined or integrated approach achieves a 99.94% detection rate or TP rate with a 0.77% FP rate. The TP rate is almost equivalent to that achieved by the search profiling approach only, while the FP rate is 36% lower. The FP rate translates to one false positive every 260 minutes, or every 4 hours and 20 minutes, as opposed to one false positive every 180 minutes or 3 hours.

Table 1. Experimental results of the search profiling and integrated modeling approaches using 2-minute quanta for feature vector extraction

Method	**True Pos. (%)**	**False Pos. (%)**
Search Profiling	100	1.12
Combined Approach	99.94	0.77

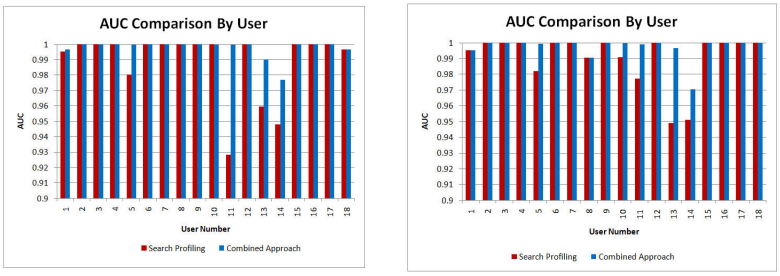
We can further reduce the frequency of false positives to one every 5 and a half hours (338 minutes), if we use the same modeling approach over 5-minute

quanta. This is derived from the 1.48 false positives recorded every 5*100=500 minutes, as reported in Table 2. While this is still a relatively high frequency of false positives, it can be further reduced if we increase the look-ahead time window where we check for decoy access. Recall that we postulated that detecting a high-volume search followed by a decoy file access corroborates the suspicion that the user is impersonating another victim user. In our current modeling scheme using 2-minute (or 5-minute) time epochs, we account only for decoy accesses that happen simultaneously with abnormal search actions, or within 2 minutes (5 minutes) at most of the high-volume search activity. If we widen this time window, we can improve the accuracy performance even further.

Table 2. Experimental results of the search profiling and integrated modeling approaches using 5-minute quanta for feature vector extraction

Method	**True Pos. (%)**	**False Pos. (%)**
Search Profiling	100	2.38
Combined Approach	100	1.48

To compare the individual classifiers for each user using the two detection schemes, we built Receiver Operating Curves (ROC) for each classifier and calculated the Area Under Curve (AUC) for each. The higher the AUC score, the better the accuracy of the classifier. Figure 3 displays the AUC scores achieved by both detection approaches by user model. The results show that each user model using the combined detection approach achieves a higher or equal AUC score, *i.e.* equal or better accuracy results than the user model based on the search profiling approach alone. The best accuracy improvements were achieved for users 5, 11, 13 and 14. These user models had the top four FP rates amongst all user models based on search profiling alone. For these specific users, the FP reduction ranged between 33% and 67% when using the combined detection approach. This confirms the efficacy of using this combined approach to limit the number of false positives and improve the accuracy of the masquerade attack dedtctor.



(a) Modeling using feature vectors per 2-minute quanta

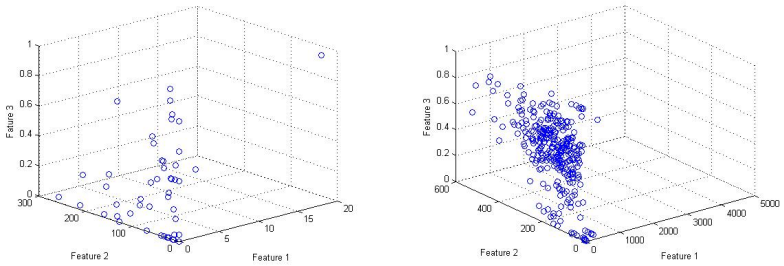
(b) Modeling using feature vectors per 5-minute quanta

Fig. 3. AUC Comparison By User Model for the Search Profiling and Intgrated Detection Approaches

5 Defending Against Mimicry Attacks

Any anomaly-based intrusion detection system (AIDS) is subject to mimicry attacks. Tan et al. [30] identified two mechanisms for performing mimicry attacks: (1) contaminating the learning and/or model update process by inserting attack data into normal user data, and (2) intertwining attacks with normal user activity so that the attacks go undetected, which is also known as an evasion attack.

We conjectured that combining the baiting technique with the user search behavior profiling technique serves as a defense mechanism against mimicry attacks, or evasion attacks in particular. We assume that user models and training data were not contaminated with masquerader data during the training or update phases. In order to demonstrate our conjecture, one would ideally have a masquerader mimic a legitimate user's behavior. However, when simulating masquerade attacks as described in our 'capture-the-flag' exercise, it was extremely difficult to make the volunteers participating in the user study mimic the behavior of a specific user. To evaluate our conjecture though, we reviewed all search behavior models and identified the user who exhibited the most similar search behavior as the search behavior of masquerade attackers. To identify this user, we measured the similarity between the legitimate user behavior and masquerader behavior by applying the probability product kernel to the distribution of their feature vectors [16]. User 13 showed the closest behavior to masqueraders as can be seen from Figure 4, which depicts the distribution of the three search-related features for user 13, and for all masqueraders combined. We can support this conjecture by reviewing the accuracy results of this user's model in Figure 3, which are indeed significantly better than the search behavior only model.



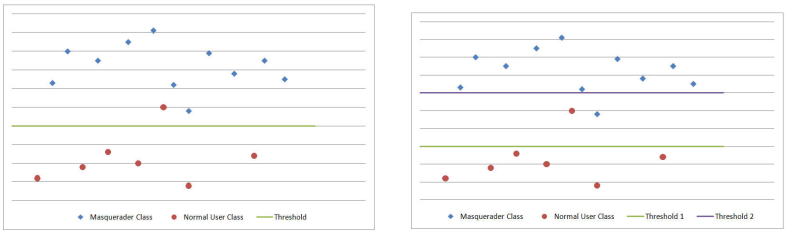
(a) Feature Vectors for User 13

(b) Feature Vectors for Masquerade Attackers

Fig. 4. Feature Vectors for User 13 and Masquerade Attackers

One might expect that hardening the detector against mimicry attacks could drive higher FP rates. Our results show the opposite. Figure 5 helps in understanding how this can be achieved. When using the search profiling approach only, the circular point above the threshold line in Figure 5(a) triggers a false positive. If we use a lower threshold beyond which search behavior is considered suspicious as in Figure 5(b), we can widen the anomaly space for the detector. This in turn means that the adversary has to work harder in order to faithfully mimic the legitimate

user's behavior. However, this alone may introduce false positives. By combining search profiling with the baiting technique, we can use a second threshold for the highly abnormal search behavior, beyond which we can achieve 100% TP rate. For points that fall in the 'ambiguous' space between the two thresholds, the access to decoy information can be used to inform the classification decision. The key to this process is the use of decoy documents that are strategically placed, highly-enticing and conspicuous in the file system, so that the attacker is very likely to touch them. Our 'Capture the-flag' exercise showed that all masqueraders did indeed touch at least one of the placed decoy files as can be seen in Figure 6[1]



(a) Search Profiling Classifier Example

(b) Search Profiling and Baiting Classifier Example

Fig. 5. Anomaly Space for the Search Profiling and the Combined Approach Classifiers

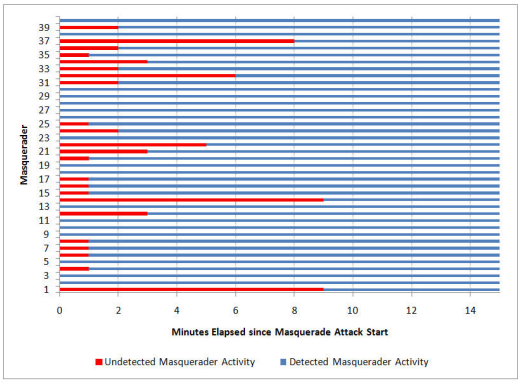


Fig. 6. Detection Time by User

[1] This figure has been published in prior work. For the purpose of preserving author anonymity, we do not refer to the paper where it was here. But we will refer to this work in the camera-ready version of this paper.

6 Conclusion

Masquerade attacks pose a serious computer security problem. Most prior work focused on profiling users. In this paper, we presented an integrated detection approach where we combine profiling user search behavior with a baiting approach based on the deployment of decoy documents on the user's file system. We reduced false positives by 36% over the best results reported in literature to date with a 99.94% masquerade detection rate with only 0.77% of false positives, the best results achieved in the literature so far. In our future work, we plan on extending this detection approach from a local file system setting to a cloud setting.

References

1. Ben-Salem, M. RUU dataset: http://www1.cs.columbia.edu/ids/RUU/data/.
2. Ben-Salem, M., Hershkop, S., and Stolfo, S. J. A survey of insider attack detection research. In *Insider Attack and Cyber Security: Beyond the Hacker* (2008), Springer.
3. Ben-Salem, M., and Stolfo, S. J. Decoy document deployment for effective masquerade attack detection.
4. Ben-Salem, M., and Stolfo, S. J. Detecting masqueraders: A comparison of one-class bag-of-words user behavior modeling techniques. In *MIST '10: Proceedings of the Second International Workshop on Managing Insider Security Threats, Morioka, Iwate, Japan* (June 2010), pp. 3–13.
5. Ben-Salem, M., and Stolfo, S. J. Modeling user search-behavior for masquerade detection. In *Columbia University Computer Science Department, Technical Report # cucs-033-10* (2010).
6. Bowen, B., and Hershkop, S. Decoy Document Distributor: http://sneakers.cs.columbia.edu/ids/ruu/dcubed/.
7. Bowen, B. M., Hershkop, S., Keromytis, A. D., and Stolfo, S. J. Baiting inside attackers using decoy documents. In *SecureComm'09: Proceedings of the 5th International ICST Conference on Security and Privacy in Communication Networks* (2009).
8. CERT. 2010 e-crimes watch survey, 2010.
9. Chang, C.-C., and Lin, C.-J. Libsvm: http://www.csie.ntu.edu.tw/ cjlin/libsvm/.
10. Chawla, N. V., Eschrich, S., and Hall, L. O. Creating ensembles of classifiers. In *Proceedings of the 2001 IEEE International Conference on Data Mining* (Washington, DC, USA, 2001), ICDM '01, IEEE Computer Society, pp. 580–581.
11. Coull, S. E., Branch, J., Szymanski, B., and Breimer, E. Intrusion detection: A bioinformatics approach. In *Proceedings of the 19th Annual Computer Security Applications Conference* (2001), pp. 24–33.
12. Dietterich, T. G. Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems* (2000), Springer-Verlag, pp. 1–15.
13. Domingos, P. Bayesian averaging of classifiers and the overfitting problem. In *Proceedings of the Seventeenth International Conference on Machine Learning* (San Francisco, CA, USA, 2000), ICML '00, Morgan Kaufmann Publishers Inc., pp. 223–230.
14. Dzeroski, S., and Zenko, B. Is combining classifiers better than selecting the best one. In *Proceedings of the Nineteenth International Conference on Machine Learning* (San Francisco, CA, USA, 2002), ICML '02, Morgan Kaufmann Publishers Inc., pp. 123–130.
15. Gashi, I., Stankovic, V., Leita, C., and Thonnard, O. An experimental study of diversity with off-the-shelf antivirus engines. In *NCA '09: Proceedings of the 2009 Eighth IEEE International Symposium on Network Computing and Applications* (Washington, DC, USA, 2009), IEEE Computer Society, pp. 4–11.

16. Jebara, T., Kondor, R., and Howard, A. Probability product kernels. *J. Mach. Learn. Res. 5* (December 2004), 819–844.
17. Krawczyk, H., Bellare, M., and Canetti, R. RFC2104, HMAC: Keyed-Hashing for Message Authentication. The Internet Engineering Task Force (IETF).
18. Littlewood, B., and Strigini, L. Redundancy and diversity in security. In *Computer Security ESORICS 2004, 9th European Symposium on Research Computer Security, LNCS 3193* (2004), Springer, pp. 423–438.
19. Maloof, M. A., and Stephens, G. D. elicit: A system for detecting insiders who violate need-to-know. In *RAID* (2007), pp. 146–166.
20. Maxion, R. A. Masquerade detection using enriched command lines. *Dependable Systems and Networks, International Conference on 0* (2003), 5.
21. Maxion, R. A., and Townsend, T. N. Masquerade detection using truncated command lines. In *DSN '02: Proceedings of the 2002 International Conference on Dependable Systems and Networks* (2002), IEEE Computer Society, pp. 219–228.
22. Maxion, R. A., and Townsend, T. N. Masquerade detection augmented with error analysis. *IEEE Transactions on Reliability 53*, 1 (2004), 124–147.
23. Maybury, M., Chase, P., Cheikes, B., Brackney, D., Matzner, S., Hetherington, T., Wood, B., Sibley, C., Marin, J., and Longstaff, T. Analysis and detection of malicious insiders. In *Proceedings of the International Conference on Intelligence Analysis* (2005).
24. Mitra, S., Saxena, N. R., and McCluskey, E. J. A design diversity metric and analysis of redundant systems. *IEEE Trans. Comput. 51*, 5 (2002), 498–510.
25. Oka, M., Oyama, Y., Abe, H., and Kato, K. Anomaly detection using layered networks based on eigen co-occurrence matrix. In *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection* (2004).
26. Oka, M., Oyama, Y., and Kato, K. Eigen co-occurrence matrix method for masquerade detection. In *Publications of the Japan Society for Software Science and Technology* (2004).
27. Pettey, C. Gartner Says Number of Identity Theft Victims Has Increased More Than 50 Percent Since 2003: http://www.gartner.com/it/page.jsp?id=501912, 3 2007.
28. Schonlau, M., Dumouchel, W., Ju, W., Karr, A. F., Theus, M., and Vardi, Y. Computer intrusion: Detecting masquerades. *Statistical Science 16* (2001), 58–74.
29. Spitzner, L. Honeypots: Catching the insider threat. *Annual Computer Security Applications Conference* (2003).
30. Tan, K., Killourhy, K., and Maxion, R. Undermining an anomaly-based intrusion detection system using common exploits. In *Recent Advances in Intrusion Detection* (2002), LNCS, Springer, pp. 54–73.
31. Tan, K. M., and Maxion, R. A. The effects of algorithmic diversity on anomaly detector performance. In *Dependable Systems and Networks, 2005. DSN 2005. Proceedings. International Conference on* (June 2005), pp. 216 – 225.
32. Wang, K., and Stolfo, S. J. One-class training for masquerade detection. In *Proceedings of the 3rd IEEE Workshop on Data Mining for Computer Security* (2003).
33. Yuill, J., Denning, D., and Feer, F. Using Deception to Hide Things from Hackers: Processes, Principles, and Techniques. *Journal of Information Warfare 5* (2006), 26–40.
34. Yuill, J., Zappe, M., Denning, D., and Feer, F. Honeyfiles: deceptive files for intrusion detection. In *Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC* (June 2004), pp. 116 – 122.
35. Yung, K. H. Using self-consistent naïve bayes to detect masqueraders. In *PAKDD'08: Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2004), pp. 329–340.