

Model Redundancy vs. Intrusion Detection

Zhuowei Li, Amitabha Das, and Sabu Emmanuel

School of Computer Engineering, Nanyang Technological University,
50, Nanyang Avenue, Singapore 639798
zhwei.li@pmail.ntu.edu.sg
{asadas, asemmanuel}@ntu.edu.sg

Abstract. A major problem faced by intrusion detection is the intensive computation in the detection phase, and a possible solution is to reduce model redundancy, and thus economize the detection computation. However, the existing literature lacks any formal evaluation of the significance of model redundancy for intrusion detection. In this paper, we try to do such an evaluation. First, in a general intrusion detection methodology, the model redundancy in the behavior model can be reduced using feature ranking and the proposed concept of ‘*variable-length signature*’. Then, the detection performance of the behavior model before and after model redundancy is compared. The preliminary experimental results show that the model redundancy in the behavior model is useful to detect novel intrusions, but the model redundancy due to the overlapping distinguishability among features is insignificant for intrusion detection.

1 Introduction

Intrusion detection becomes more and more indispensable among the security infrastructures to protect our computing resources. Basically, the task of intrusion detection is to detect intrusions from the audit trails left by computing resources. However, with the increasing bandwidth of the Internet and great volume of electronic transactions, the audit trails left for intrusion detection are also increasing. To make matters worse, most intrusion detection techniques are computation intensive, especially for anomaly-based intrusion detection (Debar et al. [1]). Thus, it is a critical issue to decrease the computation in the detection phase for intrusion detection.

In this paper, we will address *the issue of the computational cost of the detection phase* and explore ways to reduce it. In general, the intensive computation can be mitigated if the redundant information in the behavior model is reduced. For example, if we can determine an intrusion (e.g., *portsweep*) with only two features, additional features for intrusion detection other than the two features are redundant, and they will cause additional computation in the detection phase. Thus, it is feasible to lessen the computational intensity by reducing the model redundancy. However, in the past literature, there is no formal evaluation on whether reduced model redundancy will affect the efficiency of the

behavior model for intrusion detection. Based on a formal methodology, we do such evaluation in this paper.

The main contributions of this paper are as follows. First, the features are ranked for intrusion detection using a formal methodology, and some features are discarded if they do not contribute to the distinguishability of the behavior model. Then, the concept of variable-length signature is introduced to further reduce model redundancy. Lastly, based on the ranked features and variable-length signatures, a framework is designed to evaluate the model redundancy. The threat to variable-length signatures from mimicry attacks is also analyzed.

The remaining parts are organized as follows. In Section 2, a methodology for intrusion detection is described briefly, and the variable-length signature is introduced. The model redundancy in the behavior model is discussed in Section 3, and a mechanism is designed to reduce the model redundancy. In Section 4, we describe the experiments performed to evaluate model redundancy. The related works are discussed in Section 5, and we conclude the paper in the last section.

2 A Formal Intrusion Detection Methodology

In general, the behavior models of the resources for intrusion detection are built using a set of features, or a *feature vector* $FV = \{F_1, F_2, \dots, F_n\}$, where F_i is a nominal, discrete or continuous feature. Based on whether it is found in normal and/or intrusive audit trails, a feature value of any feature in the feature vector is labeled as *normal*, *suspicious* or *anomalous*. More specifically, if a feature value occurs only in the normal audit trails, it is a *normal* feature value. If it occurs only in the intrusive audit trails, for example, in the signatures of SID, it is labeled as *anomalous*. Otherwise, i.e., if it occurs in the normal and intrusive audit trails, it is labeled as *suspicious*. For brevity, we refer to the normal, suspicious, or anomalous label as the **NSA label** of a feature value.

2.1 Feature Ranges

In the feature space of discrete/continuous features, the range of values between any two feature values will be called a *feature range*. The concept of NSA labels (i.e., normal, suspicious and anomalous) can be extended to feature ranges as follows. For continuous and discrete features, if two neighboring (c.f. Section 3) feature values have identical NSA label, then the intervening range will also have the same NSA label as the two end points. This label will be valid as long as no known feature value emerges with a different NSA label within that range. On the other hand, if two neighboring feature values have different NSA labels, then the intervening range will be divided into two subranges with respect to a user-define strategy, and each subrange will be assigned the NSA label of the original feature value associated with that subrange. For nominal features, the concept of feature range is not applicable, but, for the sake of uniformity of description, each nominal feature value is also referred to as a feature range with its corresponding NSA label.

Based on the NSA labels of feature ranges, the feature space $Def(F)$ can be split into feature subranges $\{R_F^1, R_F^2, \dots, R_F^m\}$, such that the neighboring feature ranges (such as R_F^j and R_F^{j+1}) have different NSA labels. Finally, by grouping the feature ranges with identical NSA labels, we can partition the feature space into three feature subspaces: normal, suspicious and anomalous. We will denote the normal feature subspace of a feature F as $N(F)$, in which all its feature ranges come from the normal audit trails. Its suspicious and anomalous feature subspaces are denoted as $S(F)$ and $A(F)$ respectively. Thus,

$$\begin{aligned} N(F) &= \{R_F^j \mid 1 \leq j \leq m, \text{label}(R_F^j) = \text{'normal'}\} \\ S(F) &= \{R_F^j \mid 1 \leq j \leq m, \text{label}(R_F^j) = \text{'suspicious'}\} \\ A(F) &= \{R_F^j \mid 1 \leq j \leq m_i, \text{label}(R_F^j) = \text{'anomalous'}\} \end{aligned}$$

where, $label(\dots)$ returns the NSA label of a feature range. In the following description, we denote $\Omega(F) = N(F) \cup S(F) \cup A(F)$.

Example 1. Let us assume that there are three features F_1, F_2 and F_3 , $FV = \{F_1, F_2, F_3\}$. The example feature instances are listed in Table 1, in which ‘A’, ‘B’, ‘C’ and ‘D’ represent four different intrusions, and ‘N’ represents normal behaviors. The feature ranges for every feature are listed in (b).

Table 1. A simple example

(a) Example instances.				(b) Example feature ranges.			
F_1	F_2	F_3	STATUS	FEATURE INDEX	FEATURE RANGES	STATUSES	NSA LABEL
TCP	1	0.01	N	F_1	$R_{F_1}^1$	TCP	N,A ‘suspicious’
ICMP	2	0.04	N		$R_{F_1}^2$	ICMP	N ‘normal’
NETBIOS	6	0.10	C		$R_{F_1}^3$	UDP	N,B,D ‘suspicious’
TCP	4	0.08	A		$R_{F_1}^4$	NETBIOS	N,C ‘suspicious’
UDP	5	0.06	B	F_2	$R_{F_2}^1$	[1,2]	N ‘normal’
UDP	8	0.14	D		$R_{F_2}^2$	[3,5]	A,B ‘anomalous’
NETBIOS	6	0.10	N		$R_{F_2}^3$	[6,6]	N,C ‘suspicious’
UDP	7	0.02	N		$R_{F_2}^4$	[7,7]	N ‘normal’
UDP	8	0.14	B		$R_{F_2}^5$	[8,8]	B,D ‘anomalous’
				F_3	$R_{F_3}^1$	[0.01,0.05)	N ‘normal’
					$R_{F_3}^2$	[0.05,0.09)	A,B ‘anomalous’
					$R_{F_3}^3$	[0.09,0.14]	N,C,D ‘suspicious’

2.2 Compound Features

The concept of NSA labels and subspace partitioning can easily be extended to more than one feature. For convenience, we will refer to a single feature as an *atomic feature* and a combination of multiple features as a *compound feature*. We formally define a compound feature as follows:

Definition 1 (compound feature). *For any two features F_1 and F_2 , a compound feature F_{12} is defined as a subset of the cartesian product of $\Omega(F_1)$ and $\Omega(F_2)$, such that each element in this set actually represents some audit trails.*

In other words, if the ordered pair $(a, b) \in \Omega(F_{12})$, then the compound feature range (a, b) must identify some training audit trails.

$$\Omega(F_{12}) = \{(a, b) | a \in \Omega(F_1), b \in \Omega(F_2), (a, b) \text{ identifies some audit trails}\}$$

For convenience, $F_{12} = F_1 \times F_2$.

Example 2. Using the example instances in Section 2.1, for $F_{23} = F_2 \times F_3$,

- $R_{F_{23}}^1 = R_{F_2}^1 \times R_{F_3}^1, N \Rightarrow \text{'normal'}$;
- $R_{F_{23}}^2 = R_{F_2}^2 \times R_{F_3}^2, A, B \Rightarrow \text{'anomalous'}$;
- $R_{F_{23}}^3 = R_{F_2}^3 \times R_{F_3}^3, N, C \Rightarrow \text{'suspicious'}$;
- $R_{F_{23}}^4 = R_{F_2}^4 \times R_{F_3}^1, N \Rightarrow \text{'normal'}$;
- $R_{F_{23}}^5 = R_{F_2}^5 \times R_{F_3}^3, B, D \Rightarrow \text{'anomalous'}$;

and, for $F_{123} = F_1 \times F_{23}$ (i.e., the signature base in USAID[3]),

- $R_{F_{123}}^1 = R_{F_1}^1 \times R_{F_{23}}^1, N \Rightarrow \text{'normal'}$;
- $R_{F_{123}}^2 = R_{F_1}^2 \times R_{F_{23}}^1, N \Rightarrow \text{'normal'}$;
- $R_{F_{123}}^3 = R_{F_1}^1 \times R_{F_{23}}^2, A \Rightarrow \text{'anomalous'}$;
- $R_{F_{123}}^4 = R_{F_1}^3 \times R_{F_{23}}^2, B \Rightarrow \text{'anomalous'}$;
- $R_{F_{123}}^5 = R_{F_1}^4 \times R_{F_{23}}^3, N, C \Rightarrow \text{'suspicious'}$;
- $R_{F_{123}}^6 = R_{F_1}^3 \times R_{F_{23}}^4, N \Rightarrow \text{'normal'}$;
- $R_{F_{123}}^7 = R_{F_1}^3 \times R_{F_{23}}^5, B, D \Rightarrow \text{'anomalous'}$;

Intuitively, similar to atomic features, the feature ranges of the new compound feature F_{12} can also be labeled as normal, suspicious and anomalous ones, i.e., they also have the NSA labels. Furthermore, just like the atomic feature space, the feature space of a compound feature can also be partitioned into three feature subspaces, i.e., $\Omega(F_{12}) = N(F_{12}) \cup S(F_{12}) \cup A(F_{12})$.

Since the compound feature built from two atomic features shows the same property as any of its component atomic feature, it can be treated like an atomic feature to build higher order compound features. Using this recursive procedure, the feature vector FV can be converted into an equivalent n -order compound feature $F_{1\dots n}$ with subspaces $N(F_{1\dots n})$, $S(F_{1\dots n})$ and $A(F_{1\dots n})$.

2.3 Variable-Length Signatures

As stated above, for *any* (atomic/compound) feature, there are normal, suspicious and anomalous feature ranges in its feature space. In other words, the feature can distinguish some behaviors falling into normal and anomalous feature ranges as normal or anomalous, which is the task of intrusion detection. Therefore, in intrusion detection, the special characteristic of the feature can be utilized to reduce the number of features in the detection phase.

Definition 2 (Signature). *Assuming that there exists a feature vector $FV = \{F_1, F_2, \dots, F_n\}$, the feature ranges of every feature are determined beforehand. A signature is a feature range of $F_{1\dots n}$. In other words, the signature is the combination of feature ranges of all features in the feature vector.*

Example 3. In the example of Section 2.1, one signature is $(ICMP, [1, 2], [0.01, 0.05])$. The total number of possible signatures in this example is $4*5*3=60$.

Definition 3 (Variable-Length Signature). *Given a feature vector $FV = \{F_1, F_2, \dots, F_n\}$ with labelled feature ranges for every feature, a variable-length signature is a feature range that is constituted by the first i features ($i \leq n$) from FV , such that its NSA label is either 'normal' or 'anomalous'. The signature is called variable as the number of features involved in it can vary from 1 to n .*

Example 4. As in above Example 3, for the same signature $(ICMP, [1, 2], [0.01, 0.05])$, considering that the NSA label of the feature range $(ICMP)$ is 'normal', $(ICMP)$ is a 'normal' variable-length signature. $(TCP, [3, 5])$ is also an 'anomalous' variable-length signatures of the intrusion 'A'.

Among the relations between signatures and variable-length signatures, a 'normal' or 'anomalous' signature will include at least one variable-length signature, but a 'suspicious' signature does not include any variable-length signature. Thus, any variable-length signature reduces this redundancy in its corresponding signature without sacrificing the distinguishability.

3 Model Redundancy in the Behavior Model

In the behavior model, *model redundancy* is the behavior information that can be discarded without loss of signature distinguishability. There are two ways to reduce model redundancy of the behavior model in our methodology. First, due to the overlapping distinguishability among features, some features can be discarded in all signatures. Secondly, for some signatures, some features can be further discarded to form a variable-length signatures.

3.1 A Mechanism for Reducing Model Redundancy

Roughly speaking, the model redundancy in signatures can be reduced as follows. First the significance of every feature is evaluated and a sorted feature vector is identified from the original feature vector. Using the sorted feature vector, the signature building procedure constructs a so-called NSA signature base, in which the model redundancy is reduced. Lastly, a detection mechanism is designed to evaluate the usefulness of model redundancy in the behavior model. In addition, two labelled datasets are needed in our evaluation: a training audit trails and a test audit trails. The two datasets are labelled correctly with corresponding statuses (e.g., 'normal', 'Nimda', 'MSBlast').

3.2 Feature Subspaces Extraction

Step 1: feature value collection. In the labeled training audit trails, the feature values are collected for every feature. The statuses (*normal and/or intrusions*) of every feature value are also collected and stored in its status list.

Based on its status list, every feature value is assigned an NSA label. Note that this step is applied to nominal, discrete and continuous features, but the following two steps are only applicable to discrete and continuous features.

Step 2: feature value clustering. The objective of this step is to form initial feature ranges for every feature by clustering the neighboring feature values. For a discrete feature, two feature values x_1 and x_2 are *neighboring* if $|x_1 - x_2| = 1$ ¹. For a continuous feature, two feature values x_1 and x_2 are neighboring if $|x_1 - x_2| \leq \delta$. If several neighboring feature values have the same NSA label, they will be combined to form an *initial feature range*. As a special case, if, for a feature value, its neighboring feature values have different NSA labels from itself, it forms an initial feature range whose upper and lower bounds are both equal to the feature value itself. Every initial feature range thus formed inherits the NSA label of the feature values falling within it.

Step 3: feature range generalization. However, under most scenarios, the initial feature ranges of a feature will not cover all of its feature space. Any feature subspace outside the labelled feature ranges is named as an *uncovered subspace*. Comparing to the neighboring definition of feature values, the neighboring of feature ranges is defined as follows. Two feature ranges are *neighboring* if there is no other feature range between them. Thus, the uncovered subspace between any two neighboring feature ranges is processed as follows: if the two feature ranges have the same NSA label, a new feature range will be formed to cover the two feature ranges as well as the uncovered subspace; otherwise, the uncovered space is divided and allocated to these two defined feature ranges using a *predefined splitting strategy*. The NSA labels of the initial feature ranges will be inherited by the newly extended or combined feature ranges.

In this paper, an uncovered space will be shared by its neighbors equally. Ultimately, all the *known* feature space of every feature will be covered by well-defined feature ranges.

3.3 A Sorted Feature Vector

Notations. Given a compound feature $F_{1\dots i}$, whose normal, suspicious and anomalous feature subspaces are $N(F_{1\dots i})$, $S(F_{1\dots i})$ and $A(F_{1\dots i})$ respectively, and an atomic feature F_{i+1} . Basically, it is possible that a feature range labelled ‘anomalous’ is caused by several intrusions, but the feature range due to a single intrusion is extremely valuable for identifying that intrusion, and is thus highly desirable. Therefore, $A(F_{1\dots i})$ is further split into two feature subspaces $An(F_{1\dots i})$ and $A1(F_{1\dots i})$. Specifically, if the feature range in $A(F_{1\dots i})$ is caused by only one intrusion, it will be classified as $A1(F_{1\dots i})$, otherwise, as $An(F_{1\dots i})$. In summary, there are four feature subspaces for $F_{1\dots i}$: $N(F_{1\dots i})$, $S(F_{1\dots i})$, $An(F_{1\dots i})$, and $A1(F_{1\dots i})$, and for $F_{1\dots i+1}$: $N(F_{1\dots i+1})$, $S(F_{1\dots i+1})$, $An(F_{1\dots i+1})$, and $A1(F_{1\dots i+1})$ respectively.

¹ If the distance is larger than 1, there is an uncovered space $|x_1 - x_2 - 1|$.

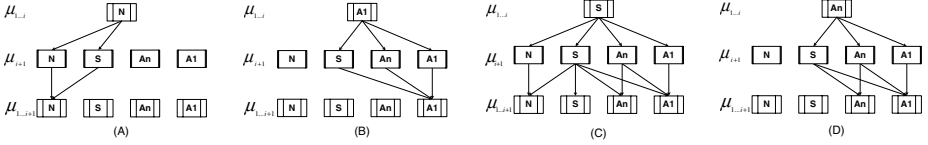


Fig. 1. Four compounding scenarios

Measuring and Sorting Features. In Figure 1, the four compounding scenarios are illustrated to design a measure for feature selection. As indicated in (A), the compounding between $N(F_{1...i})$ and feature subspaces of F_{i+1} will result in feature ranges in $N(F_{1...i+1})$. Thus, there is no increment of the distinguishability for intrusion detection. Similarly, (B) will not lead to any benefit either. In contrast, the compounding scenarios (C) and (D) will increase the distinguishing ability of $F_{1...i+1}$ as some of their resultant feature ranges fall within $N(F_{1...i+1})$ or $A1(F_{1...i+1})$. As a result, to increase the distinguishability of $F_{1...i+1}$, the increment due to compounding scenarios (C) and (D) should be maximized.

Considering that our ultimate objective is to reduce the computation overhead in the detection phase, it is useful to consider frequency information of every feature range. For this reason, we assume that the training audit trials have the same frequency distribution to the test audit trails, thus the detection computation will be reduced. With respect to the compounding operation between $F_{1...i}$ and F_{i+1} , the following notations are defined.

- $\#S2N(F_{1...i}, F_{i+1})$ is the number of instances falling into $S(F_{1...i})$ and $N(F_{1...i+1})$.
 - $\#S21(F_{1...i}, F_{i+1})$ is the number of instances falling into $S(F_{1...i})$ and $A1(F_{1...i+1})$.
 - $\#n21(F_{1...i}, F_{i+1})$ is the number of instances falling into $An(F_{1...i})$ and $A1(F_{1...i+1})$.
- $newIdentify(F_{i+1}|F_{1...i}) = \#S2N(F_{1...i}, F_{i+1}) + \#S21(F_{1...i}, F_{i+1}) + \#n21(F_{1...i}, F_{i+1})$

The feature evaluation (Algorithm 1) works as follows. Initially, FV_{sort} is empty. In i -th step, all features outside FV_{sort} are evaluated by $newIdentify$, and the feature with the maximal value will be appended to FV_{sort} as the last element. Obviously, the feature with maximal distinguishability is selected.

Algorithm 1 Feature sorting and selecting for intrusion detection

Require: Feature Vector FV , Number of Features n , Training audit trails Σ_{tr} ;
 1: Selecting one feature F_{max} with the maximal size of instances in Σ_{tr} falling into $N(F_k)$ and $A1(F_k)$ ($1 \leq k \leq n$);
 2: Inserting F_{max} into FV_{sort} ;
 3: **for** $i = 2$ to n **do**
 4: $maxIdentify = -\infty$; $maxIdentifyFeature = -1$;
 5: **for** $j = 1$ to n **do**
 6: **if** $F_j \in FV_{sort}$ **then** continue;
 7: Building the compound feature by compounding F_j and the features in FV_{sort} ;
 8: Calculating $newIdentify(F_j|FV_{sort})$ from Σ_{tr} ;
 9: **if** $newIdentify > maxIdentify$ **then** $maxIdentify = newIdentify$; $maxIdentifyFeature = j$; **endif**
 10: **end for**
 11: Inserting $F_{maxIdentifyFeature}$ into FV_{sort} ;
 12: **end for**
 13: Output FV_{sort} ;

3.4 Building NSA Signature Base

The NSA signature base building process starts with the first feature in FV_{sort} , and includes one additional feature in each step in the sorted order. Finally, after compounding with the last feature, the compounding feature ranges in $S(F_{FV_{sort}})$ and $An(F_{FV_{sort}})$ are left as signatures in the signature base. Algorithm 2 describes it in detail.

Algorithm 2 Building NSA signature base

Require: FV_{sort} and its size m , Training audit trails;
1: NSABase= Φ ;
2: **for** $i = 1$ to m **do**
3: Building the compound feature $F_{1\dots i}$ in FV_{sort} ;
4: **for** each feature range in $N(F_{1\dots i})$ and $A1(F_{1\dots i})$ **do**
5: **if** it does not match the signatures in NSABase **then**
6: Inserting it into NSABase as a variable-length signature;
7: **end if**
8: **end for**
9: **end for**
10: Inserting $S(F_{1\dots m})$ and $An(F_{1\dots m})$ to NSABase;
11: Output NSABase;

Table 2. Several example entries in the NSA signature base

VARIABLE-LENGTH SIGNATURES	NSA	INTRUSIONS OR NORMAL
[94,A]	A	portsweep
[1069,N]	N	normal
[2,S]+[214,S]	A	buffer_overflow
[0,S]+[182,S]+[84,N]	N	normal
[0,S]+[86,S]+[0,S]	A	smurf
[0,S]+[277,S]+[239,S]+[14,S]	A	back

Table 2 shows several example entries in the NSA signature base. In every entry, the feature range is denoted by its index, and its NSA label is paired with the index using a pair of square brackets. The compounding of the feature ranges is denoted by ‘+’. For instance, in ‘[2,S]+[214,S]’, ‘[2,S]’ refers to the second range of the first feature in FV_{sort} (which is ‘source bytes’, please see FV_{sort} in the appendix), and ‘S’ indicates that its NSA label is ‘suspicious’. The whole expression, along with its NSA label and status, constitutes a complete variable-length signature “[2,S]+[214,S]=>A:buffer_overflow”.

Obviously, if there exist ‘suspicious’ signatures in the NSA signature base (i.e., $S(F_{1\dots m}) \neq \Phi$ in Algorithm 2), it implies that the feature vector is not enough to completely distinguish the intrusive and normal behaviors in the audit trails. Because of the ambiguity in the suspicious signatures, they will lead to detection errors (*false alarms or false negatives*) for intrusion detection.

Definition 4 (Signature Variation). *In the signature base of a computing resource, if (variable-length) signatures A and B have the same status list, A is called the signature variation of B, and vice versa.*

It is obvious that the signature variations of an intrusion indicate the existence of intrusion variations, which lead to the well-known detection difficulty in SID techniques. In general, the number of signature variations in the signature base indicates the diversity of the corresponding intrusions.

3.5 Detection Mechanisms via Variable-Length Signatures

In the detection phase, the feature ranges of every feature in the sorted feature vector and the NSA signature base are used to detect anomalies in the test or real-time audit trails. First, the feature instance is extracted from the audit trails. Then, a corresponding variable-length signature is generated incrementally by mapping of the feature values one at a time. The signature thus generated is compared with the variable-length signature base, and if a match is found, the instance is identified by the matched entry and the remaining features are not used. If there is no such match using all features, the instance is ‘*anomalous*’.

3.6 Mimicry Attacks

The threat from mimicry attacks (introduced by Wagner et al.[6]) can be explained as follows. In USAID[3], a mimicry attack achieve its goal by mimicking all the feature ranges of a ‘normal’ signature. In contrast, a mimicry attack to variable-length signatures can be designed more easily than USAID since the short ‘normal’ variable-length signatures will be the preference to attackers. In other words, variable-length signatures become more vulnerable than USAID.

4 Experiments

We have chosen a typical dataset for network intrusion detection from KDD CUP 1999 contest, in which every record is an instance of a specific feature vector collected from the audit trails. This is because the dataset meets the requirements of our methodology: *labeled audit trails and intrusion-specific feature vector*. For a detailed description of the datasets, please refer to ‘<http://www-cse.ucsd.edu/users/elkan/clresults.html>’. As the precision of a continuous feature is 0.01 in the dataset, we let $\delta = 0.01$.

4.1 Experimental Results

In our following sections, we will focus on mining the sorted feature vector, building the NSA signature base, and evaluating the model redundancy. To save space, we would refer the reader to [3] for the details of the feature ranges.

Mining a Sorted Feature Vector. Figure 2 illustrates the percentage of identified instances in the audit trails with more features being added into the sorted feature vector. Obviously, the curve converges to a stable value as more features

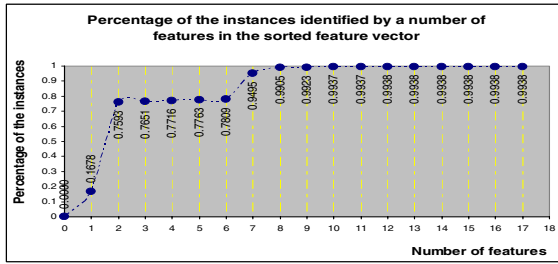


Fig. 2. The distinguishability tendency in mining the sorted feature vector

are included into the sorted feature vector, i.e., more instances in the training audit trails are identified by these features. The stopping criterion for feature selection is defined when, for any feature F_k in our applied feature vector, $newIdentify(F_k|F_{FV_{sorted}}) = 0$. That is, there are no instances in the training audit trails will be identified by adding a further feature. Using this criterion, (m=)17 features are selected into the sorted feature vector, i.e., (n-m=)24 features are discarded before continuing with further experiments (for details, c.f. the sorted feature vector in the appendix).

Building the NSA Signature Base. With respect to the sorted feature vector and the feature ranges for every feature, every instance in the labeled audit trails is evaluated as a variable-length signature. Then, these variable-length signatures

Table 3. The statistics of NSA signature base

ITEM	$N(\dots)$	$S(\dots)$	$An(\dots)$	$AI(\dots)$	TOTAL
Size of NSA base with 41 features	60371	43	15	2779	63208
Size of NSA base with variable features	952	43	15	1072	2082
Average length of variable-length signatures	5.79	17	17	5.53	5.97

constitute the NSA signature base, and thus the model redundancy is reduced in comparison with the NSA signature base with constant-length signatures. Table 3 lists the statistics of the variable-length signatures in the NSA signature base. In summary, the storage space of NSA signature base using variable-length signatures (i.e., the model redundancy) is cut down by 99.52% in comparison with constant-length signatures. At the same time, the non-empty signature set $S(\dots)$ and $An(\dots)$ indicates that the information in the feature vector is not enough to distinguish all the behaviors in the training audit trails.

Table 4 summarizes the signature variations in the NSA signature base. It is clear that the same intrusion can lead to multiple variable-length signatures, and the number of these signature variations indicate the diversity of the intrusion and thus the difficulty to detect these intrusions. At the same time, it is worth noting that the most diverse behavior is the ‘normal’ behavior, which has the largest number of ‘normal’ signature variations. This phenomenon can

Table 4. Number of signature variations in the training audit trails

NAME / LENGTH	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
normal	141	214	28	29	62	37	67	4	76	265	1	17	5	2	0	0	8
neptune	1	0	23	45	0	147	3	43	9	0	0	7	3	0	1	1	0
buffer-overflow	1	10	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0
portsweep	2	0	84	48	7	66	2	37	0	2	0	0	0	0	0	1	6
warezclient	3	19	0	1	0	2	7	0	3	37	1	0	0	0	0	0	0
back	19	8	3	10	3	2	0	0	12	0	0	5	0	0	0	0	0
pod	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
teardrop	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rootkit	0	2	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0
smurf	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
phf	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
warezmaster	0	1	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0
multihop	0	4	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0
satan	0	8	19	102	1	43	1	12	1	13	2	0	0	0	0	0	0
imap	0	3	0	1	0	1	0	0	2	0	0	0	0	0	0	0	0
ftp-write	0	5	0	0	0	0	1	0	0	2	0	0	0	0	0	0	0
ipsweep	0	1	2	1	6	12	20	0	2	8	0	0	0	0	0	0	0
nmap	0	0	52	0	0	1	3	0	1	1	0	0	1	0	0	0	0
guess-passwd	0	0	1	1	8	0	0	0	0	1	0	0	0	0	0	0	0
land	0	0	0	1	0	0	3	0	0	0	0	0	1	0	2	0	0
load module	0	0	0	0	0	2	1	0	0	6	0	0	0	0	0	0	0
spy	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
perl	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0

be explained as follows. In the normal usage of a computing resource, there are many tasks to do with different objectives, and different tasks lead to different signatures. However, for an intrusion, the objective of the task is simple comparatively to attack and destroy a computing resource, so the intrusive task is not complicated as normal tasks. Therefore, an intrusion behavior will cause fewer signatures than ‘normal’ behavior.

Evaluating Model Redundancy. To evaluate the significance of model redundancy for intrusion detection, three experiments are done using *constant-length signatures with 41 features (SIG-41)*, *constant-length signatures with the selected 17 features (SIG-17)*, and *variable-length sigantures with the sorted feature vector (varSIG)* respectively. In SIG-41, all the model redundancy is included in the NSA signature base, but some model redundancy due to overlapping distinguishability among features are eliminated in SIG-17, and lastly, the model redundancy in the signatures are further eliminated in varSIG. In Table 5², they are compared with respect to four efficiency parameters: (1) FAR: false alarm rate; (2) DR-Known: detection rate for known intrusions; (3) DR-New: detection rate for new intrusions; (4) Num-FR: number of feature ranges in detecting an instance.

Table 5. Efficiency parameters for intrusion detection

PARAMETERS	FAR (%)	DR-Known (%)	DR-New (%)	Num-FR
SIG-41	1.451	99.779	98.184	41
SIG-17	1.409	99.778	97.901	17
varSIG	0.431	95.928	55.172	3.205

² Similar to USAID with constant-length sigantures [3], two new intrusions, namely, snmpgetattack and mailbomb, are eliminated from the detection results.

In Table 5, the negligible difference between detection efficiency (i.e., the same detection rate and false alarm rate) for SIG-41 and SIG-17 indicates that our feature selection is effective, and that the model redundancy due to the overlapping distinguishability among features can be discarded. Even though the detection rate of varSIG is lower than SIG-41 and SIG-17, it is still encouraging because it achieves much lower false alarm rate and significantly lights computation overhead (i.e., smaller Num-FR).

The reason for lower detection rate in varSIG can be explained as follows. For every constant-length signature in the NSA signature base, there is one corresponding variable-length signature, in which some feature ranges are discarded for compactness. However, some intrusions will only cause anomalies in the discarded feature ranges. For example, assuming that a constant-length signature is “[0,N]+[14,S]+[23,S]=>N:Normal” and the corresponding variable-length signature is “[0,N]=>N:Normal”. If the signature of an instance is “[0,N]+[25,S]+[23,S]”, it will be detected as ‘normal’ by the variable-length signature but as ‘anomaly’ by the constant-length signature. For this reason, although it leads to intensive computation in the detection phase, the model redundancy in a signature, which can be reduced in a variable-length signature, is critical to enhance the efficiency in detecting novel intrusions. Conversely, the significance of model redundancy is rooted in the incompleteness of the behavior model (e.g., novel intrusions).

5 Related Work

Even though feature selection is well known as a critical step for intrusion detection [2], most research on AID techniques utilize expert knowledge to select the features manually [4], and there are only a few reported work relating to feature evaluation for intrusion detection [5] [2]. In [5], feature ranking is done by evaluating the whole performance for intrusion detection. In our methodology, feature evaluation is done by maximizing the feature distinguishability between normal and anomalous signatures only in the training phase. In MADAM ID [2], the statistical patterns are first mined from the network audit data, and then these patterns are used as guidelines to select features for intrusion detection. Unfortunately, the statistical patterns may not reflect the audit trails completely, and there is no guarantee that all feature distinguishability will be used in the detection phase. However, in this paper, as the stopping criteria for the feature selection is that the distinguishing ability will not change even with more features, all feature distinguishability in the feature vector will be included in the sorted feature vector.

6 Conclusions and Future Work

In this paper, we try to evaluate the usefulness of model redundancy in the behavior model for intrusion detection. To achieve it, the concept of variable-length signatures and a feature selection methodology are proposed to reduce the redun-

dancy in the behavior model. Our preliminary experimental results show that the model redundancy due to the overlapping distinguishability among features are insignificant for intrusion detection. Even though the model redundancy cut down by variable-length signatures is critical in detecting novel intrusions, it can lead to many benefits: lowering the false alarm rate, compacting the behavior model and significantly lightening the detection computation overhead.

There are still several problems left unsolved, and they will be addressed in our future work. (1) Adjusting the degree of redundancy in the signature to enhance the detection performance; (2) Whether the discarded features will evolve into critical ones in the future or under other environments; (3) Selecting the features considering the cost of a feature to be changed in mimicry attacks.

References

1. H. Debar, M. Dacier, and A. Wespi. A revised taxonomy for intrusion detection systems. *Annales des Telecommunications*, 55(7-8):361-378, 2000.
2. W. Lee and S.J. Stolfo. A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security*, 3(4):227-261, Nov. 2000.
3. Zhuowei Li and Amitabha Das. Unifying anomaly-based and signature-based intrusion detection. Technical Report CAIS-TR-2004-005, CAIS, Aug. 2004.
4. M.V. Mahoney and P.K. Chan. Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks. In *SIGKDD 2002*, July 23-26 2002.
5. S Mukkamala and A H. Sung. Identifying significant features for network forensic analysis using artificial intelligent techniques. *International Journal of Digital Evidence*, 1(4):1-17, Winter, 2003.
6. David Wagner and Paolo Soto. Mimicry attacks on host-based intrusion detection systems. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 255-264, 2002.

The Sorted Feature Vector in Our Experiments

$FV_{sort} = \{\text{Source Bytes, Service, Flag, REJ Error Rate, Destination Host Service REJ Error Rate, Same Service Rate, Destination Host Service Different Host Rate, Different Service Rate, Service Different Host Rate, Destination Bytes, Protocol Type, Service SYN Error Rate, Destination Host Service SYN Error Rate, Logged In, Land, SYN Error Rate, Duration}\}$.