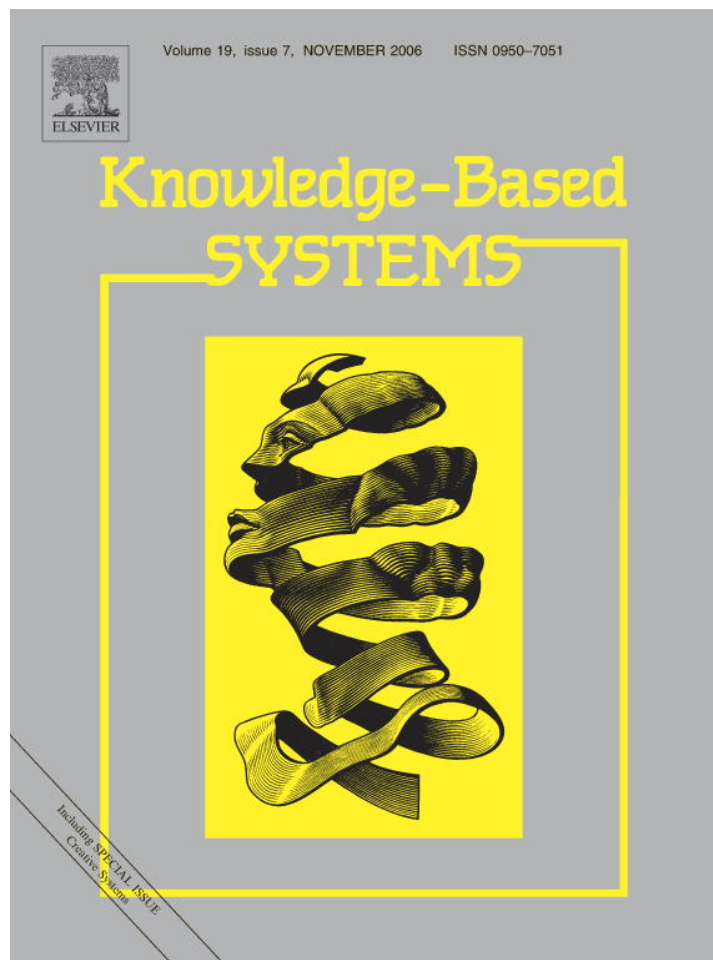


Provided for non-commercial research and educational use only.
Not for reproduction or distribution or commercial use.



This article was originally published in a journal published by Elsevier, and the attached copy is provided by Elsevier for the author's benefit and for the benefit of the author's institution, for non-commercial research and educational use including without limitation use in instruction at your institution, sending it to specific colleagues that you know, and providing a copy to your institution's administrator.

All other uses, reproduction and distribution, including without limitation commercial reprints, selling or licensing copies or access, or posting on open internet sites, your personal or institution's website or repository, are prohibited. For exceptions, permission may be sought for such use through Elsevier's permissions site at:

<http://www.elsevier.com/locate/permissionusematerial>



ELSEVIER

Available online at www.sciencedirect.com

 ScienceDirect

Knowledge-Based Systems 19 (2006) 576–591

Knowledge-Based
SYSTEMS

www.elsevier.com/locate/knosys

Analyzing and evaluating dynamics in stide performance for intrusion detection

Zhuwei Li ^{*}, Amitabha Das

School of Computer Engineering, Nanyang Technological University, 50, Nanyang Avenue, Singapore 639798, Singapore

Received 18 March 2005; accepted 10 March 2006

Available online 27 June 2006

Abstract

Anomaly-based intrusion detection (AID) techniques are useful for detecting novel intrusions into computing resources. One of simple but typical AID detectors proposed to date is stide, which is based on analysis of system call sequences. In this paper, we present a detailed formal framework to analyze, understand and improve the performance of stide and similar AID techniques. Several important properties of stide-like detectors are established through formal theorems, and validated by carefully conducted experiments using test datasets. Finally, the framework is utilized to reduce the cost of developing AID detectors by identifying the critical sections in the training dataset.

© 2006 Elsevier B.V. All rights reserved.

Keywords: Intrusion detection; Computer security; Framework; Stide; System call

1. Introduction

Since the concept of intrusion detection in computer systems was proposed by Anderson [1], many research studies have been carried out to find appropriate intrusion detection techniques to protect the resources in computers or networks [20,17,19,4]. However, the network disaster caused recently by Nimda, MSBlast and MSSasser highlights the shortcomings of the intrusion detection techniques deployed in our network infrastructures [14], and indicates that intrusion detection techniques still have a long way to go before they can provide effective protection to computing resources.

In general, the intrusion detection techniques can be categorized into *signature-based intrusion detection (SID)* and *anomaly-based intrusion detection (AID)* ones. The SID techniques build (and/or update) intrusion signature bases that include signatures of most known intrusions. Then, the resource behavior that matches any intrusion signature

in the bases is labeled as an intrusion. Obviously, previously unknown intrusions cannot be detected by this technique. Besides this drawback, the requirement of instant updating of the intrusion signature bases imposes a severe performance bottleneck on SID techniques.

For this reason, AID techniques have become a focus of intense research as they offer an useful alternative to SID techniques, that is capable of detecting novel intrusions. One of their implicit assumptions is that the violations or anomalies in the audit trails are indications of intrusions, i.e. the anomaly is caused by an intrusion into the resource. This assumption, though largely correct, may not be always true as some malicious intrusions may not violate the parameters of normal behaviors, whereas some non-malicious activities may appear as violations [25].

In principle, almost all AID techniques work as follows. First, a model of normally behaving users [12,13] and/or processes [10,19] is built. Then, an intrusion is detected by comparing the actual current behavior against the normal model and taking actions according to some pre-determined security policies [4,3,21]. Although many AID techniques have been proposed to date, no single AID technique can effectively detect all types of intrusions into the

^{*} Corresponding author.

E-mail addresses: zhwei.li@pmail.ntu.edu.sg (Z. Li), asadas@ntu.edu.sg (A. Das).

resources under various scenarios [14]. More specifically, they suffer from high false alarm rate that tends to reduce the effectiveness of true alarms because of the base rate fallacy, “that in order to achieve substantial values of the Bayesian detection rate $P(\text{Intrusion}|\text{Alarm})$, we have to achieve a (perhaps in some cases unattainably) low false alarm rate” [2]. In other words, the higher false alarm rate will make a security site officer frustrated, and then to ignore all (true and false) alarms. In addition, the high cost of and insufficient guideline about training the normal model do not make matters any better for AID.

In this paper, instead of proposing a new AID technique, we develop a formal framework to argue about and analyze the properties of a typical AID technique called *stide* [8,6]. Then, we evaluate the influence of the completeness of audit trails on the detection efficiency of *stide*. Generally speaking, to be efficient, any AID technique must try to increase the detection rate simultaneously keeping the false alarm rate to a minimum. For this reason, it is important to understand the factors that suppress the detection rate, and lead to false alarms. In this framework for *stide*, the factors are identified as the minimum foreign sequences in the intrusive dataset and the maximum self sequences in the test dataset, and the relations between these factors and *stide* efficiency are expressed and discussed. In addition, most related works to *stide* are interpreted in a logical way under this framework, namely, mimicry attacks, information hiding techniques, t-*stide*, variable-length patterns, and locality frame scheme.

Our aim in this paper is just that, and we do so by providing a useful formalism that not only helps in our understanding of the underlying dynamics among various factors (e.g. the completeness of the training dataset, the complexity of processes etc.), but also provides practical methodologies as to how to develop efficient training procedures for AID detectors to make training faster. Contradictory to our general concept that more training audit trails will lead to more efficient *stide* detectors, the experimental results show that there are critical sections in the training audit trails, which are important to *stide* efficiency. Our trimming scheme is to find such critical sections in the training audit trails. Ultimately, the framework gives guidelines for selecting *stide* for intrusion detection (i.e., what are the applicable scenarios of *stide*). Though our discussion is based on *stide*, the framework provides insights that are more generally applicable to other sequence analysis based AID techniques (but excluding ones using probability information of sequences [11,15,29]).

1.1. Related work

Sequence time-delay embedding or (*stide*), was first proposed by Forrest et al. [6] for privileged Unix processes. The method is an instance of a computer immunology system to protect the computer systems using the principles of natural immune systems. However, throughout the series of papers by Forrest et al. [10,22,27], there is this

“magic number 6” which is empirically determined to be the length of *stide* detectors to obtain effective detection of all anomalies in intrusive datasets. This so called ‘Why six?’ problem [23] for *stide* has stimulated a lot of research [18,24]. Finally, Tan et al. [24,23] showed that the correct answer to the problem lies in the fact that the lower bound on the *stide* detector length is determined by the length of the minimum foreign sequence(s) in the intrusive dataset.

As a further work to Tan et al. [24], a comprehensive framework, that systematically analyzes the interactions among various factors affecting the operational limits of *stide* detectors, is proposed in this paper. In particular, the effects of incompleteness of the training dataset on the effectiveness of *stide* are established in the framework. Though it is generally understood that more completeness of the training audit trails leads to higher detection rate and lower false alarm rate, a quantitative relationship among them is not available in the literatures. In practice, even though there exist some techniques to generate near-complete training dataset [5], it is worth studying the effect of the completeness of the training dataset on the performance of *stide* detectors for the following reasons:

- (1) It is difficult to collect a complete training dataset even with the completeness-guarantee techniques;
- (2) The existing completeness-guarantee techniques are only specific for the system-call-based events in a host. However, as a general technique, *stide* is not only applicable to system call sequences in a host, but also other event sequences in diverse environments, such as networks;
- (3) In a converse manner, the knowledge of the precise influence of the completeness of the training dataset on the efficiency of *stide* detectors will be useful to propose or improve completeness-guarantee techniques;
- (4) The ‘concept drifting’ problem in the normal model for anomaly-based intrusion detection tends to make the normal model always incomplete. For example, under the network scenario, the traffic model will change with more applications applied;
- (5) Most important of all, the tradeoff between the completeness of the training dataset and the efficiency of the *stide* detector needs to be quantified. It is a common sense that, as the training dataset approaches more completeness, more efforts are needed to achieve any information gain. Furthermore, it is possible that the efficiency loss due to the incompleteness will be made up by modeling generalization.

Contributions of this paper. The main contributions are summarized below:

- A formal framework is proposed to determine the operational limits for *stide*-like AID detectors (without using

frequency/probability information in the behavior model). Under the framework, the other techniques related to stide, namely, mimicry attacks, information hiding techniques, variable length patterns, t-stide and locality frame, are interpreted as well in a logical way.

- The influence of the completeness of the training dataset on stide efficiency is evaluated. From our experimental results, we found that stide is not appropriate for intrusion detection under dynamic scenarios, such as the Internet traffic.
- A scheme is derived from the formal framework for trimming the training data given a specific detection performance by identifying and eliminating non-critical sections since they yield no additional information gain. This saves both training time and space for storing training data.

The remaining paper is organized as follows. Section 2 gives the notations and definitions to help the readers in understanding the rest of the paper. In Section 3, stide is briefly introduced and expressed formally. The performance measures such as the effectiveness, completeness and efficiency of an anomaly-based intrusion detector are defined and several theorems on them are presented and proved in Section 4. In addition, the operational limits for stide detectors are determined. In Section 5, the influence of the completeness of training dataset on stide efficiency is evaluated. In the last section, we draw the conclusions.

2. Notations and definitions

2.1. Notations

2.1.1. Sequences and sequence sets

Let Σ denote the dataset for a process, which consists of event logs with the identity of the associated running process. A sequence S in Σ is an event series constituted by contiguous events in Σ with the same process identity, and its length is denoted as $|S|$. Specially, ϕ is a sequence with length 0, and Σ itself is a sequence as well. $SS(\Sigma, l)$ denotes the set of all the sequences of length l ($l \geq 0$), which are collected from Σ . Thus, $SS(\Sigma, 0) = \{\phi\}$. Furthermore, $SS(\Sigma) = \bigcup_{l=0}^{+\infty} SS(\Sigma, l)$. In any subset of $SS'(\Sigma) \subset SS(\Sigma)$, $|SS'|_{min}(\Sigma)$ ¹ is the minimum length of all sequences in $SS'(\Sigma)$, and $SS'_{min}(\Sigma)$ consists of the sequences with length $|SS'|_{min}(\Sigma)$ in $SS'(\Sigma)$. As a special case, $|SS'|_{min}(\Sigma) = 0$ if $\phi \in SS'_{min}(\Sigma)$, and $|SS|_{min}(\Sigma) = 1$.

Example 2.1. Suppose $\Sigma = abc$. ab is a sequence in Σ with length 2, $SS(\Sigma, 2) = \{ab, bc\}$, and $SS(\Sigma) = \{\phi, a, b, c, ab, bc, abc\}$. For a subset $SS'(\Sigma) = \{b, c, ab, abc\}$, $|SS'|_{min}(\Sigma) = 1$ and $SS'_{min}(\Sigma) = \{b, c\}$.

¹ We use the notation $|\cdot|$ to represent the length of any member sequence in a sequence set, instead of its size.

2.1.2. Set operations

For given datasets Σ_1 and Σ_2 of a process and corresponding sequence sets $SS(\Sigma_1, l)$ and $SS(\Sigma_2, l)$, the set operations ($\cup, \cap, -$) are defined as follows ($l \geq 0$):

- (1) $SS(\Sigma_1, l) \cup SS(\Sigma_2, l) = \{S | (S \in SS(\Sigma_1, l)) \vee (S \in SS(\Sigma_2, l))\}$
- (2) $SS(\Sigma_1, l) \cap SS(\Sigma_2, l) = \{S | (S \in SS(\Sigma_1, l)) \wedge (S \in SS(\Sigma_2, l))\}$
- (3) $SS(\Sigma_1, l) - SS(\Sigma_2, l) = \{S | (S \in SS(\Sigma_1, l)) \wedge (S \notin SS(\Sigma_2, l))\}$

In addition, $\Sigma_1 \odot \Sigma_2$ is a special concatenation of the datasets Σ_1 and Σ_2 , such that there is no sequence in $SS(\Sigma_1 \odot \Sigma_2, l)$, in which some events belong to Σ_1 and other events belong to Σ_2 . This is because the process identity in Σ_1 is different from that in Σ_2 . Therefore,

$$SS(\Sigma_1 \odot \Sigma_2, l) = SS(\Sigma_1, l) \cup SS(\Sigma_2, l).$$

Example 2.2. Suppose that $\Sigma_1 = abc$ and $\Sigma_2 = ab$. $SS(\Sigma_1, 2) = \{ab, bc\}$, and $SS(\Sigma_2, 2) = \{ab\}$. Thus, the set operations $SS(\Sigma_1, 2) \cup SS(\Sigma_2, 2) = \{ab, bc\}$, $SS(\Sigma_1, 2) \cap SS(\Sigma_2, 2) = \{ab\}$, and $SS(\Sigma_1, 2) - SS(\Sigma_2, 2) = \{bc\}$. $\Sigma_1 \odot \Sigma_2 = abc; ab$.

2.1.3. Supersequence and subsequence

If S_{sub} is a contiguous subsequence of S and $|S| - |S_{sub}| = k$, then S_{sub} is said to be a k -order subsequence of S , and denoted as $S_{sub} \preceq_k S$. Similarly, $S_{sup} \succeq_k S$ denotes that S_{sup} is a k -order supersequence in which S is a contiguous subsequence, and $|S_{sup}| - |S| = k$. It is worth noting that $\phi \preceq_{|SS|}$, and $S \succeq_{|S|} \phi$. For example, $ab \preceq_1 abc$, $a \preceq_2 abc$ and $ab \succeq_1 a$. In addition, the terms *subsequence* and *supersequence* will always imply contiguity in this paper, such that $ac \not\preceq_1 abc$, and $bc \preceq_2 abcd$.

2.2. Definitions

Central to our framework are the twin concepts of the *minimum foreign sequence* – **MFS**, and the *maximum self sequence* – **MSS**. Their definitions, expressions and relation are given below.

2.2.1. Foreign sequences and self sequences

Let Σ_{ref} be the *reference dataset*, and Σ_{tgt} be the *target dataset*. For any sequence $S \in SS(\Sigma_{tgt})$, if S is also in $SS(\Sigma_{ref})$, S will be called a *self sequence*, otherwise, it is a *foreign sequence* to Σ_{ref} . Furthermore, $FRGN(\Sigma_{tgt}|\Sigma_{ref})$ is defined as the set of foreign sequences of Σ_{tgt} w.r.t. Σ_{ref} . Similarly, the set of self sequences is defined as $SELF(\Sigma_{tgt}|\Sigma_{ref})$. Mathematically,

$$FRGN(\Sigma_{tgt}|\Sigma_{ref}) = \bigcup_{l=1}^{+\infty} SS(\Sigma_{tgt}, l) - SS(\Sigma_{ref}, l), \quad (1)$$

$$SELF(\Sigma_{tgt}|\Sigma_{ref}) = \bigcup_{l=1}^{+\infty} SS(\Sigma_{tgt}, l) \cap SS(\Sigma_{ref}, l). \quad (2)$$

Thus, $FRGN(\Sigma_{tgt}|\Sigma_{ref}) \cup SELF(\Sigma_{tgt}|\Sigma_{ref}) = SS(\Sigma_{tgt})$.

A sequence S in $FRGN(\Sigma_{igt}|\Sigma_{ref})$ will be called a *minimum foreign sequence (MFS)* [24] if none of its subsequences is in $FRGN(\Sigma_{igt}|\Sigma_{ref})$, i.e. all of its subsequences are in $SELF(\Sigma_{igt}|\Sigma_{ref})$. The set of all minimum foreign sequences is denoted as $MFS(\Sigma_{igt}|\Sigma_{ref})$. On the other hand, for any sequence S in $SELF(\Sigma_{igt}|\Sigma_{ref})$, if there exists one first-order supersequence that is not included in $SELF(\Sigma_{igt}|\Sigma_{ref})$ (i.e., it is in $FRGN(\Sigma_{igt}|\Sigma_{ref})$), then it will be called a *maximum self sequence (MSS)*. The set of all maximum self sequences is denoted as $MSS(\Sigma_{igt}|\Sigma_{ref})$. Formally, they can be expressed as

$$MFS(\Sigma_{igt}|\Sigma_{ref}) = \{S|\forall S'(S' \in FRGN(\Sigma_{igt}|\Sigma_{ref})) \wedge (\forall S''\forall k(S'' \in SS(\Sigma_{igt})) \wedge (S'' \preceq_k SS' \notin FRGN(\Sigma_{igt}|\Sigma_{ref})))\}, \quad (3)$$

$$MSS(\Sigma_{igt}|\Sigma_{ref}) = \{S|\forall S'(S' \in SELF(\Sigma_{igt}|\Sigma_{ref})) \wedge (\exists S''(S'' \in SS(\Sigma_{igt})) \wedge (S'' \succ_1 S) \wedge (S'' \notin SELF(\Sigma_{igt}|\Sigma_{ref})))\}. \quad (4)$$

From these definitions, $MFS(\Sigma_{igt}|\Sigma_{ref}) \subset SS(\Sigma_{igt})$ and $MSS(\Sigma_{igt}|\Sigma_{ref}) \subset SS(\Sigma_{igt})$. Furthermore, based on above notations, $|MFS|_{min}(\Sigma_{igt}|\Sigma_{ref}) \geq 1$, $|MSS|_{min}(\Sigma_{igt}|\Sigma_{ref}) \geq 0$. Specially, if $MFS(\Sigma_{igt}|\Sigma_{ref}) = \Phi$, $|MFS|_{min}(\Sigma_{igt}|\Sigma_{ref}) = +\infty$. The same property can be applied to $MSS(\Sigma_{igt}|\Sigma_{ref})$.

Example 2.3. Suppose that $\Sigma_{ref} = abc, \Sigma_{igt} = abaa$. The sequence sets of these two datasets are: $SS(\Sigma_{ref}) = \{\phi, a, b, c, ab, bc, abc\}$, $SS(\Sigma_{igt}) = \{\phi, a, b, ab, ba, aa, aba, baa, abaa\}$. Next,

$$FRGN(\Sigma_{igt}|\Sigma_{ref}) = \{ba, aa, aba, baa, abaa\}$$

$$SELF(\Sigma_{igt}|\Sigma_{ref}) = \{\phi, a, b, ab\}.$$

Finally, we can deduce:

$$MFS(\Sigma_{igt}|\Sigma_{ref}) = \{ba, aa\}$$

$$MSS(\Sigma_{igt}|\Sigma_{ref}) = \{a, b, ab\}$$

$$MFS_{min}(\Sigma_{igt}|\Sigma_{ref}) = \{ba, aa\}$$

$$MSS_{min}(\Sigma_{igt}|\Sigma_{ref}) = \{a, b\}$$

$$|MFS|_{min}(\Sigma_{igt}|\Sigma_{ref}) = 2$$

$$|MSS|_{min}(\Sigma_{igt}|\Sigma_{ref}) = 1$$

2.2.2. Relation between MFS and MSS

One relationship between MFSs and MSSs of two datasets Σ_{ref} and Σ_{igt} is given by the following theorem.²

Theorem 2.4. For two datasets Σ_{ref} and Σ_{igt} of a process ($SS(\Sigma_{ref}) \neq SS(\Sigma_{igt})$), the following relation holds.

$$|MSS|_{min}(\Sigma_{igt}|\Sigma_{ref}) = |MFS|_{min}(\Sigma_{igt}|\Sigma_{ref}) - 1. \quad (5)$$

Specially, if $SS(\Sigma_{ref}) = SS(\Sigma_{igt})$, $|MSS|_{min}(\Sigma_{igt}|\Sigma_{ref}) = |MFS|_{min}(\Sigma_{igt}|\Sigma_{ref}) = +\infty$.

Example 2.5. In Example 2.3, it is obvious,

$$|MSS|_{min}(\Sigma_{igt}|\Sigma_{ref}) = |MFS|_{min}(\Sigma_{igt}|\Sigma_{ref}) - 1 = 1.$$

3. A formal description of stide

In the experimental setup for stide [6,27], there are two datasets for every process, the normal dataset Σ_{nml} , and the intrusive dataset Σ_{int} , which are defined below.

Definition 3.1. (Normal Dataset) The normal dataset is a dataset Σ_{nml} that is utilized to train the normal model of a process for stide, and it *MUST* be collected in the normal run of the process without any intrusion.

Definition 3.2. (Intrusive Dataset) The intrusive dataset Σ_{int} is a dataset that is collected when one or more intrusions were occurring during the runs of a process.

In terms of these two datasets from the same process, stide can be formally described as follows. Let $\omega (\geq 1)$ denote the size of the detector window. In the modeling phase, the normal model of the process is obtained as: $SS(\Sigma_{nml}, \omega)$. Then, in the detecting phase, the foreign sequences in the intrusive dataset Σ_{int} , $FS(\Sigma_{int}|\Sigma_{nml}, \omega)$, are enumerated:

$$FS(\Sigma_{int}|\Sigma_{nml}, \omega) = SS(\Sigma_{int}, \omega) - SS(\Sigma_{nml}, \omega).$$

If $FS(\Sigma_{int}|\Sigma_{nml}, \omega) \neq \Phi$, the intrusion(s) in the intrusive dataset Σ_{int} can be detected with the detector length ω [10,9,24,27].³ It is evident that the sequence set $FS(\Sigma_{int}|\Sigma_{nml}, \omega)$ is strongly related to $MFS(\Sigma_{int}|\Sigma_{nml})$ via $|MFS|_{min}(\Sigma_{int}|\Sigma_{nml})$:

$$|MFS|_{min}(\Sigma_{int}|\Sigma_{nml}) \leq \omega \iff FS(\Sigma_{int}|\Sigma_{nml}, \omega) \neq \Phi. \quad (6)$$

In its formal proposal [10], a Locality Frame Count (LFC) function is applied to smooth the noise, or to filter the false alarms in the process by summing up the number of foreign sequences found within the span of a locality frame. However, the LFC function does not add to or compensate for the detecting ability, or failure/shortcoming of the stide detector, and it will not be used in our framework. As an application of our framework, it will be interpreted later.

Practically, even though the underlying principle is very simple, stide can detect most of the intrusions into the processes (*the datasets from UNM* [7]). For this reason, it is accepted as a typical and effective anomaly-based intrusion detector in many research studies.

³ It is notable that most of these research studies only apply stide to system-call based sequences in a host as does the original proposal for stide[10]. However, in principle, stide is applicable to other environments as well. Therefore, in our formal framework, it will not be specific for any environment, which is also one of our objectives to formalize the stide technique.

² All proofs of theorems in this paper are provided in the Appendix A.

4. A formal framework for stide

In general, the efficiency of an intrusion detection technique is determined by both false positives [2] and false negatives (i.e., the detection rate). The incompleteness of the normal model is well-known as the main cause for the false positives in an AID detector [16,2]. As indicated in the first section, such completeness of the normal dataset is difficult to verify, and there is no effective method to guarantee it. However, in most of the research studies on stide [6,24], the completeness is not adequately considered when evaluating the efficiency of stide detectors.

In this paper, we try to build a methodology to analyze the influence of the completeness of the normal dataset on the efficiency of stide detectors. To achieve it, the normal dataset is regarded as the training dataset Σ_{trn} to build the known normal model of the resource. At the same time, a test dataset Σ_{tst} is introduced to evaluate the completeness of the training dataset Σ_{trn} . The function of the test dataset is to evaluate the ability of the detector to correctly identify normal data as such without generating false positives. Thus, the test dataset must be collected during a normal run of a process without any intrusion as well. To some extent, our methodology corresponds to the actual scenarios where it is difficult to collect all the normal behaviors of a computing resource, and there are always false positives when the normal behaviors are examined by an AID detector. In addition, without loss of generality, we assume that the audit trails in Σ_{int} is caused by only one intrusion.

4.1. A critical look at stide performance

In our formal framework for stide, with the detector window size ω , the normal model $SS(\Sigma_{trn}, \omega)$ is first built from Σ_{trn} . Based on its detection results on Σ_{tst} and Σ_{int} , all the sequences are classified as follows. The outcome of a detection process can be divided into four categories depending on the true nature of the data and the correctness of the detection result. These are shown in Table 1. Therefore, according to whether a sequence matches the normal model $SS(\Sigma_{trn}, \omega)$, $SS(\Sigma_{tst}, \omega)$ can be split into two subsets: False Positive Sequence Set (denoted as $FPSS(\Sigma_{tst}|\Sigma_{trn}, \omega)$), and True Negative Sequence Set (denoted as $TNSS(\Sigma_{tst}|\Sigma_{trn}, \omega)$). Similarly, depending on the detection outcome, the intrusive sequence set $SS(\Sigma_{int}, \omega)$ can be split into two subsets: False Negative Sequence Set (denoted as $FNSS(\Sigma_{int}|\Sigma_{trn}, \omega)$), and True Positive Sequence Set (denoted as $TPSS(\Sigma_{int}|\Sigma_{trn}, \omega)$). Using our earlier nota-

tions, we can write the following definitions of the above four sequence subsets:

$$FPSS(\Sigma_{tst}|\Sigma_{trn}, \omega) = SS(\Sigma_{tst}, \omega) - SS(\Sigma_{trn}, \omega)$$

$$TNSS(\Sigma_{tst}|\Sigma_{trn}, \omega) = SS(\Sigma_{tst}, \omega) \cap SS(\Sigma_{trn}, \omega)$$

$$TPSS(\Sigma_{int}|\Sigma_{trn}, \omega) = SS(\Sigma_{int}, \omega) - SS(\Sigma_{trn}, \omega)$$

$$FNSS(\Sigma_{int}|\Sigma_{trn}, \omega) = SS(\Sigma_{int}, \omega) \cap SS(\Sigma_{trn}, \omega)$$

Furthermore,

$$FRGN(\Sigma_{int}|\Sigma_{trn}) = \bigcup_{\omega=1}^{+\infty} TPSS(\Sigma_{int}|\Sigma_{trn}, \omega)$$

$$SELF(\Sigma_{int}|\Sigma_{trn}) = \bigcup_{\omega=1}^{+\infty} FNSS(\Sigma_{int}|\Sigma_{trn}, \omega)$$

$$FRGN(\Sigma_{tst}|\Sigma_{trn}) = \bigcup_{\omega=1}^{+\infty} FPSS(\Sigma_{tst}|\Sigma_{trn}, \omega)$$

$$SELF(\Sigma_{tst}|\Sigma_{trn}) = \bigcup_{\omega=1}^{+\infty} TNSS(\Sigma_{tst}|\Sigma_{trn}, \omega)$$

Next, according to the sequences in these four categories, we will define two aspects of stide performance, namely effectiveness and completeness. Finally, we will give the definitions and conditions for an efficient stide detector.

4.1.1. Effectiveness of a stide detector

Definition 4.1. (Effectiveness) A stide detector with detector window ω is *effective* to detect the intrusion in Σ_{int} if there is at least one sequence in the intrusive sequence set $SS(\Sigma_{int}, \omega)$, which is detected as a true positive, i.e., $TPSS(\Sigma_{int}|\Sigma_{trn}, \omega) \neq \Phi$.

To detect an intrusion effectively, the relation between stide detector window size ω and the intrusion characteristics is critical to choose a proper ω for stide, and it is stated in the following theorem.

Theorem 4.2. *Let us assume that there are a training dataset Σ_{trn} and an intrusive dataset Σ_{int} of a process. A stide detector of length ω , built from Σ_{trn} , is effective w.r.t. Σ_{int} , iff*

$$\omega \geq |MFS|_{min}(\Sigma_{int}|\Sigma_{trn}). \quad (7)$$

Example 4.3. Suppose that $\Sigma_{trn} = aba$, and $\Sigma_{int} = ababa$. Then, $MFS_{min}(\Sigma_{int}|\Sigma_{trn}) = \{bab\}$, and $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) = 3$. As $SS(\Sigma_{int}, 1) = SS(\Sigma_{trn}, 1) = \{a, b\}$, and $SS(\Sigma_{int}, 2) = SS(\Sigma_{trn}, 2) = \{ab, ba\}$, $TPSS(\Sigma_{int}|\Sigma_{trn}, 1) = \Phi$ and $TPSS(\Sigma_{int}|\Sigma_{trn}, 2) = \Phi$. But $TPSS(\Sigma_{int}|\Sigma_{trn}, 3) = \{bab\} \neq \Phi$. Thus, only if $\omega \geq 3$, the intrusion in Σ_{int} will be detected by stide effectively.

Note that Theorem 4.2 merely summarizes the conclusion of Tan et al. [24], but in our framework, it is rather straightforward to prove its validity.

4.1.2. Completeness of a stide detector

Definition 4.4. (Completeness) A stide detector with detector window ω is *complete* if the underlying normal model built from a training dataset Σ_{trn} is complete. In other words, the sequence subsets $TNSS(\Sigma_{tst}|\Sigma_{trn}, \omega) = SS(\Sigma_{tst}, \omega)$, and thus $FPSS(\Sigma_{tst}|\Sigma_{trn}, \omega) = \Phi$.

Table 1
Four detection scenarios

	Intrusive sequence	Normal sequence
Alarm	True positive	False positive
Non-alarm	False negative	True negative

Due to the base-rate fallacy [2], the completeness of a stide detector is also critical for its application. The following theorem establishes the conditions for the completeness of a stide detector in terms of the detector window size ω .

Theorem 4.5. *Let us assume that there are a training dataset Σ_{trn} and a test dataset Σ_{tst} of a process. A stide detector of length ω , built from Σ_{trn} , is complete w.r.t. Σ_{tst} , iff*

$$\omega \leq |MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}). \quad (8)$$

Example 4.6. Suppose that $\Sigma_{trn} = aba$, and $\Sigma_{tst} = baba$. Then, $MSS_{min}(\Sigma_{tst}|\Sigma_{trn}) = \{ba, ab\}$, and $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}) = 2$. As $SS(\Sigma_{tst}, 1) = SS(\Sigma_{trn}, 1) = \{a, b\}$, and $SS(\Sigma_{tst}, 2) = SS(\Sigma_{trn}, 2) = \{ab, ba\}$, $FPSS(\Sigma_{tst}|\Sigma_{trn}, 1) = \Phi$, and $FPSS(\Sigma_{tst}|\Sigma_{trn}, 2) = \Phi$. On the other hand, $FPSS(\Sigma_{tst}|\Sigma_{trn}, 3) = \{bab\} \neq \Phi$. Thus, only if $\omega \leq 2$, the stide detector built from Σ_{trn} is complete w.r.t. Σ_{tst} .

Corollary 4.7. *For a training dataset Σ_{trn} and a test dataset Σ_{tst} of a process, if $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}) = 0$, there are no complete stide detectors built from Σ_{trn} w.r.t. Σ_{tst} .*

4.1.3. Efficient stide detectors

Definition 4.8. (Efficiency) A stide detector with detector window size ω is *efficient* w.r.t. Σ_{tst} and Σ_{int} if it is effective to detect the intrusion in Σ_{int} , and it is complete in detecting Σ_{tst} .

It is easy to conclude that an efficient stide detector will not produce any false positives when analyzing Σ_{tst} , and it will produce true positives when analyzing Σ_{int} . We are now in a position to state the condition for a stide detector to be efficient, which is expressed by the following theorem.

Theorem 4.9. *Given a training dataset Σ_{trn} , a test dataset Σ_{tst} , and an intrusive dataset Σ_{int} , a stide detector with the detection window size ω , obtained using Σ_{trn} , is efficient w.r.t. Σ_{tst} and Σ_{int} iff*

$$|MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) \leq \omega \leq |MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}) \quad (9)$$

Proof 4.10. It can be inferred from Theorems 4.2 and 4.5.

Example 4.11. Suppose that $\Sigma_{trn} = aba$, $\Sigma_{tst} = baba$, and $\Sigma_{int} = abc$. Then, $MSS_{min}(\Sigma_{tst}|\Sigma_{trn}) = \{ba, ab\}$, and $MFS(\Sigma_{int}|\Sigma_{trn}) = \{c\}$, thus $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}) = 2$ and $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) = 1$. For a stide detector with length ω , to be complete, $\omega \leq 2$, and to be effective, $\omega \geq 1$. Finally, we can get $1 \leq \omega \leq 2$.

Fig. 1 shows the area determined by $MFS_{min}(\Sigma_{int}|\Sigma_{trn})$ and $MSS_{min}(\Sigma_{tst}|\Sigma_{trn})$ where efficient stide detectors must belong. If the point determined by $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn})$ and $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn})$ is in the efficient area, it is possible to find one or more efficient stide detector(s). Otherwise, no

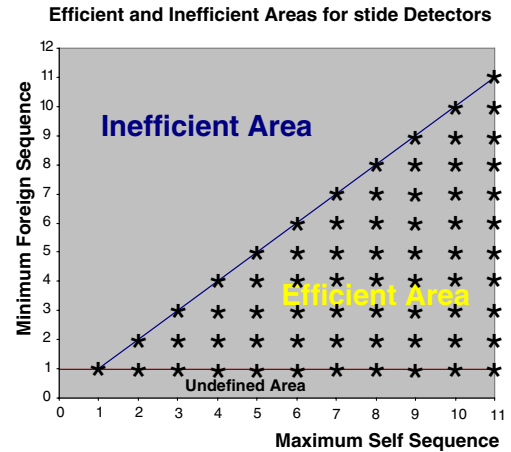


Fig. 1. Efficient and inefficient areas for stide detectors on the training dataset Σ_{trn} .

efficient stide detector can be found. Note that, there is one undefined area since $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) \geq 1$.

The following corollary, drawn from the above theorem, explicitly defines the operational limits of a stide detector.

Corollary 4.12. *For a training dataset Σ_{trn} , a test dataset Σ_{tst} , and an intrusive dataset Σ_{int} of a process, the following hold:*

- (a) If $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}) < |MFS|_{min}(\Sigma_{int}|\Sigma_{trn})$, there are no efficient stide detectors w.r.t. Σ_{tst} and Σ_{int} .
- (b) With a detector window ω , if $\omega \geq |MFS|_{min}(\Sigma_{int}|\Sigma_{trn})$, and $\omega \geq |MSS|_{min}(\Sigma_{tst}|\Sigma_{trn})$, the stide detector built by Σ_{trn} is effective, but not efficient w.r.t. Σ_{tst} and Σ_{int} .
- (c) With a detector window ω , if $\omega \leq |MFS|_{min}(\Sigma_{int}|\Sigma_{trn})$, and $\omega \leq |MSS|_{min}(\Sigma_{tst}|\Sigma_{trn})$, the stide detector built by Σ_{trn} is complete, but not efficient w.r.t. Σ_{tst} and Σ_{int} .

4.2. Completeness of the training dataset vs. stide efficiency

From their definitions, $MFS(\Sigma_{int}|\Sigma_{trn})$ and $MSS(\Sigma_{tst}|\Sigma_{trn})$ will be affected by the completeness of the training dataset to a large extent. Therefore, according to Theorem 4.9, the completeness of the training dataset is critical to stide efficiency. In Fig. 2, the universe of all possible sequences is referred to as U , in which the known and unknown normal models are only complementary parts of the complete normal model. Outside the complete normal model is the intrusive behavior space for the known and unknown intrusions. However, in stide, all the sequences from the training dataset are regarded as normal (in the known normal model), and other sequences lying outside the training dataset are considered anomalous. Obviously, the unknown normal model is critical for its efficiency. Thus, in our following analysis, we assume that the test dataset Σ_{tst} incorporates the whole unknown normal model (Fig. 2). Even though the assumption can not be achieved

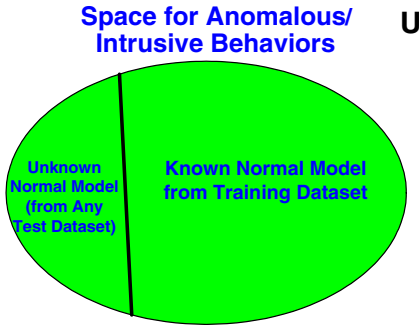


Fig. 2. Behavior spaces for *stide*: the known and unknown normal behavior models, and the intrusive behavior space.

in the real deployment, it is reasonable in analyzing *stide* performance for intrusion detection theoretically. Given this framework, let's examine the scenario in which *stide* is suitable for detecting the intrusions into a resource.

4.2.1. MSSs in the test dataset

Based on Eq. (3), $MSS(\Sigma_{tst}|\Sigma_{trn})$ is deduced as

$$\begin{aligned} MSS(\Sigma_{tst}|\Sigma_{trn}) &= \{S|\forall S(S \in SELF(\Sigma_{tst}|\Sigma_{trn})) \wedge (\exists S'(S' \in SS(\Sigma_{tst})) \wedge (S' \succ_1 S) \wedge (S' \notin SELF(\Sigma_{tst}|\Sigma_{trn})))\} \\ &= \{S|\forall S(S \in SELF(\Sigma_{tst}|\Sigma_{trn})) \wedge (\exists S'(S' \in SS(\Sigma_{tst})) \wedge (S' \succ_1 S) \wedge (S' \in FRGN(\Sigma_{tst}|\Sigma_{trn})))\}. \end{aligned}$$

Thus, $MSS(\Sigma_{tst}|\Sigma_{trn})$ is affected by $FRGN(\Sigma_{tst}|\Sigma_{trn})$, which is in the unknown normal model (Fig. 2). Theoretically, for any Σ_{trn} and Σ_{tst} , if Σ_{trn} is not complete, $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn})$ can vary from 0 to $+\infty$ (Note: $+\infty$ here indicates a potentially large number bounded by the length of the dataset Σ_{tst}).

4.2.2. MFSs in the intrusive dataset

Let's first define one more concept in our framework:

Definition 4.13. The common false positive sequence set in the intrusive dataset $CFPS(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn})$ is

$$CFPS(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) = FRGN(\Sigma_{tst}|\Sigma_{trn}) \cap SS(\Sigma_{int}).$$

It is obvious that $CFPS(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) \subset SS(\Sigma_{int})$.

Example 4.14. Suppose that $\Sigma_{trn} = ljk$, $\Sigma_{tst} = jkl$ and $\Sigma_{int} = ckl$. Based on sequence set definition, $SS(\Sigma_{trn}) = \{\phi, l, j, k, lj, jk, ljk\}$, $SS(\Sigma_{tst}) = \{\phi, j, k, l, jk, kl, jkl\}$, and $SS(\Sigma_{int}) = \{\phi, c, k, l, ck, kl, ckl\}$. Next, we get $FRGN(\Sigma_{tst}|\Sigma_{trn}) = \{kl, jkl\}$. Therefore,

$$\begin{aligned} CFPS(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) &= \{kl\} \\ |CFPS|_{min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) &= 2. \end{aligned}$$

The following theorem is deduced to determine what affects $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn})$ in our framework.

Theorem 4.15. For the datasets Σ_{trn} , Σ_{tst} , and Σ_{int} ,

$$\begin{aligned} |MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) &= \min(|CFPS|_{min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}), \\ |MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})). \end{aligned} \quad (10)$$

Example 4.16. Take the same scenario in Example 4.14. In it, $MFS_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) = \{c\}$, and $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) = 1$. Thus, we can determine that $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) = \min\{2, 1\} = 1$, which is correct as $MFS_{min}(\Sigma_{int}|\Sigma_{trn}) = \{c, kl\}$. On the other hand, if $\Sigma_{int} = jkl$, $SS(\Sigma_{int}) = SS(\Sigma_{tst})$. Thus, $CFPS(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) = \{kl, jkl\}$, $MFS_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) = \Phi$. As $MFS_{min}(\Sigma_{int}|\Sigma_{trn}) = \{kl\}$, $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) = 2 = \min(2, +\infty)$.

As indicated by the intrusive space in Fig. 2, $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$ reflects the intrusion characteristics in a specific intrusive dataset Σ_{int} , which does not depend on the completeness of the training dataset. Thus, without regard to the completeness of Σ_{trn} , $MFS(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$ and $SS(\Sigma_{int})$ will always be stable. According to Theorem 4.15, if $|CFPS|_{min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) < |MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$, $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn})$ will be affected through the set $FRGN(\Sigma_{tst}|\Sigma_{trn})$ as the completeness of the training dataset increases.

In summary, both $MFS(\Sigma_{int}|\Sigma_{trn})$ and $MSS(\Sigma_{tst}|\Sigma_{trn})$ are affected by the completeness of the training dataset Σ_{trn} , i.e. $FRGN(\Sigma_{tst}|\Sigma_{trn})$.

4.2.3. Enhancing efficiency of a stide detector

Theorem 4.17. Assume that, for a process, the training dataset from which the known model is built, is Σ_{trn} , the test dataset from which the whole unknown normal model is built, is Σ_{tst} , and the intrusive dataset is Σ_{int} . Then, there exist one or more efficient *stide* detectors iff

$$|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}) \geq |MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}). \quad (11)$$

Example 4.18. Take the two scenarios in Examples 4.14 and 4.16. If $\Sigma_{int} = ckl$, $MSS_{min}(\Sigma_{tst}|\Sigma_{trn}) = \{l, k\}$, $MFS_{min}(\Sigma_{int}|\Sigma_{trn}) = \{c\}$, and $MFS_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) = \{c\}$. Thus, there exists only one efficient *stide* detector with length $\omega = 1$. However, if $\Sigma_{int} = jkl$, $MSS_{min}(\Sigma_{tst}|\Sigma_{trn}) = \{l, k\}$, $MFS_{min}(\Sigma_{int}|\Sigma_{trn}) = \{kl\}$, and thus, there do not exist efficient *stide* detectors. At the same time, as $MFS_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) = \phi$, and $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) = +\infty$, the above equation in Theorem 4.17 does not hold.

What the above theorem tells us is that with increasing completeness of the training dataset, the intrusion characteristics reflected by $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$ acts more and more as a threshold for the efficiency of a *stide* detector. Therefore, sooner or later, the intrusion must manifest itself in the intrusive dataset with a finite (reasonably small) length of the MFSs, otherwise, there will be no efficient *stide* detector for the intrusion.

The following corollary, which follows from the Theorems 4.17 and 4.9, emphasizes the condition to build efficient *stide* detectors from a training dataset:

Corollary 4.19. Assume that, for a process, the training dataset from which the known normal model is built, is Σ_{trn} , the test dataset from which the unknown normal model is

constructed, is Σ_{tst} , and the intrusive dataset is Σ_{int} . Then, if $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn}) < |MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$, there are no efficient stide detectors.

Ideally, if the training dataset Σ_{trn} is complete so that it includes all the normal behaviors of a process (i.e., for any Σ_{tst} , $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn}) = +\infty$, $|CFPS|_{min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) = +\infty$. At the same time, the MFSs in an intrusive dataset Σ_{int} , is in fact $MFS(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$, which is the absolutely ideal scenario. Definitely, under the ideal scenario, there will be efficient stide detectors trained by the dataset Σ_{trn} .

4.3. Interpretation of related work on stide

Following the publication of stide, several research studies have been published with criticisms and suggestions of improvement of stide [28,25,27,26]. Under our proposed framework, they can be interpreted in a logical way to determine their basic foundations.

4.3.1. Mimicry attacks and intrusion information hiding

The mimicry attacks are proposed by Wagner to show the weakness of the stide technique [28]. In a nutshell, the proposed strategies to do mimicry attacks are: *intrusive behavior avoidance; waiting for the intrusive behaviors accepted by normal model passively and actively; replacing the system call parameters; inserting no-effect system calls; creating equivalent variations of a given malicious sequence*. Almost with the same principles, the information hiding paradigm is also applied to indicate that the stide technique is easy to be evaded [25].

Utilizing our proposed framework, it is very obvious that all these evading strategies are to make the minimum foreign sequences of an intrusion as large as possible so that eventually it grows beyond the length of stide detector window, making the stide detector ineffective. From the viewpoint of information theory, the information gain in the intrusive dataset (which is manipulated by mimicry attacks or information hiding techniques) is too small to be detected. Furthermore, since these two techniques focus on applying stide to system call sequences in the host-based systems with more or less strong assumptions, the only conclusion that can be made is that the stide is not suitable for detecting the mimicry attacks based on the system call sequences. Therefore, at this point, no conclusion can be drawn about the influence of these techniques on the efficiency of a general stide detector (with a ‘good’ encoding replacing ‘system call’).

In addition, we found that it is possible to make stide inefficient by getting control of one application and then nudging it to generate smaller *minimum common false positives* during the run of an intrusion (Theorem 4.15). If the quantity of the false positives are large enough during the intrusion, the stide detector will be useless in detecting the intrusion due to the base-rate fallacy [2].

4.3.2. t-stide and variable length patterns

As variations of stide, t-stide and variable-length patterns are proposed in [27] and [26], and both of them utilize the frequency information of each sequence, t-stide is very similar to stide except that it discards infrequent sequences whose frequency is smaller than a threshold t [27]. The performance of t-stide is found to be unsatisfactory by the author. Using our framework, the obvious reason for t-stide’s failure is that the discarded sequences will increase the incompleteness of the training dataset, that will decrease its detection efficiency. The ultimate reason is identified as a design drawback in t-stide [16].

Since the principles for stide and variable-length patterns [26] are different, their comparisons will be based on the detection performance by considering the fact that only the patterns (or sequences) are used in the detection phase. As indicated in our framework, the minimum foreign sequence is the main characteristic left in the audit trails for the sequence-related AID techniques. In the principles of variable-length patterns, we noticed that if the minimum length of the MFSs of an intrusion is larger than 1, the intrusion will be easily ignored by variable-length patterns. For example, suppose that the normal model for the variable-length method is {ABCD, CAE, FBD}. If the minimum foreign sequence of an intrusion is ‘DC’, and the intrusive audit trail is ‘ABCDCAEFBD’, the intrusion will not be detected. As the experimental results in [26] are very good, we suspect that the MFS of the chosen intrusions is 1, just like the ‘misconfiguration’ for ‘wu-ftpd’ in our experiments described later.

4.4. The significance of locality frame count

For stide, following [27,10], the anomaly value of a trace is derived from the number of mismatches occurring in a temporally local region, called a locality frame (LF). Then, a locality frame count (LFC) is used as a threshold to determine whether the locality frame is anomalous in the trace.

Let us assume that the stide detector window is of length ω . Suppose that, in a locality frame of a trace, there are at least n MFSs: $MFS_1, MFS_2, \dots, MFS_n$ with lengths smaller than ω , and MFS_k has the minimum length l_k among them. Since by definition, one MFS can not completely include another MFS, the minimum number of mismatches will take place when the MFS’s are maximally overlapped, i.e., MFS_2 starts one event later than MFS_1 , MFS_3 starts one event later than MFS_2 and so on. In this pathological case, the minimum number of anomalies detected in the LF should be $\omega - l_k + n$. Therefore, for successful detection of the anomaly in the LF, we must have $LFC \leq \omega - l_k + n$.

From the above analysis, we can identify these ways to successfully detect intrusions in an LF: (1) making the detector window larger; (2) making the minimum foreign

sequence smaller, and (3) making the number of MFSs in one locality frame as large as possible to form a cluster of anomalies [10]. Since larger detector window will degrade the efficiency of stide, the latter two options can be considered to serve as a guideline for choosing proper lengths for LF and LFC.

5. An Application of the framework

Apart from strengthening the comprehension of the inherent dynamics of stide-like AID detectors, the formal framework can also be applied to accelerate the training of stide-like AID detectors with less training audit trails. In this section, other than evaluating the influence of the completeness of training dataset on stide efficiency, an application will be described in detail: trimming the normal dataset without losing efficiency for a given detection performance (thus the training procedure is speed up).

5.1. Experimental setup and datasets

For the convenience of comparison, the datasets [7] that are used in [27,24] have been used in our experiments as well. In addition, we have discarded the normal datasets of several processes that are too small to use in our framework. The normal and intrusive datasets for selected processes are specified in Table 2.

From the table, our selected datasets represent most processes and intrusions into the processes. Furthermore, to analyze the characteristics of every intrusion, its intrusive dataset, even into the same process, is treated as an individual dataset.

Table 2
The dataset specifications

Normal datasets	Intrusive datasets	No. of traces	No. of system calls
Live-named-UNM	–	142	9230572
–	Buffer overflow-1	3	969
–	Buffer overflow-2	2	831
Live-lpr-MIT	–	2703	2926304
–	Lprcp	1001	165248
Sendmail-CERT	–	294	1576086
–	Syslog-local-1	6	1516
–	Syslog-local-2	6	1574
–	Syslog-remote-1	7	1861
–	Syslog-remote-2	4	1553
–	Cert-sm565a	3	275
–	Cert-sm5x	8	1537
Sendmail-UNM	–	346	1799764
–	Decode	36	3067
–	Forward loops	36	2569
–	Sunsendmailcp	3	1119
Syn-wu-ftpd	–	8	180315
–	Misconfiguration	5	1363
Syn-xlock-UNM	–	71	339177
–	Buffer overflow-1	1	489
–	Buffer overflow-2	1	460

5.2. The completeness of training dataset vs. stide efficiency

Before applying an AID technique for intrusion detection, it is critical to evaluate what are its applicable scenarios since any AID technique is not a panacea within the context of intrusion detection. In this section, the influence of the completeness of the training dataset on the efficiency of stide detectors will be evaluated to achieve two objectives: (1) *evaluating the relation between the completeness of the training dataset and stide efficiency to get what are the applicable scenarios of stide*, and (2) *trimming the redundancy in the training dataset with sacrificing the distinguishability of the behavior model*. For that purpose, we regard the normal dataset for every process to be complete for the normal model of the process, and we induce incompleteness by splitting the normal dataset into a training dataset and a test dataset. To remove any dependency, we choose the training datasets with m varying sizes $Size_1, \dots, Size_m$ and n varying starting points Pos_1, \dots, Pos_n within the length of the normal dataset Σ_{nml} . To achieve it, the normal dataset is treated as a continuous ring using wrap around of the linear dataset. Given any splitting point Pos_i and any size $Size_j$, the part from Pos_i to $(Pos_i + Size_j) \% |\Sigma_{nml}|$ is selected as $\Sigma_{trn}(i, j)$, and whatever remains is chosen as the test dataset $\Sigma_{tst}(i, j)$. Based on $\Sigma_{trn}(i, j)$ and $\Sigma_{tst}(i, j)$, the completeness of the training dataset will be evaluated considering stide efficiency in the detection phase.

On the other hand, in stide-like AID techniques, the frequency (or probability) information of the events in the training dataset is not utilized, so trimming the repeated events (or sequences) in the training dataset is useful to economize the training time without any loss of efficiency. In the trimming procedure, the *critical sections* in a normal dataset are identified to produce a compact training dataset. One of the requirements for the critical section is that the stide detectors trained by it must be as efficient as when they are trained by the complete untrimmed dataset. Finally, the most compact critical section(s) in the normal dataset are chosen for stide without sacrificing its efficiency.

To achieve it, we also develop two graphical tools to make it easy and convenient to analyze the completeness of the training dataset, and the characteristics of the datasets. They are described below.

5.2.1. MFS-MSS average curves

These curves are inspired by Theorem 4.17, which can be used to depict the influence of the completeness of the training dataset on the detection efficiency graphically. At the same time, our objective for these curves is to evaluate the dynamics in stide efficiency with the completeness of training dataset, thus, we only concern about the size of training dataset. For this reason, the average values for $|MSS|_{min}(\Sigma_{tst}|\Sigma_{trn})$ and $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn})$ for a given training data size $Size_j$ ($1 \leq m$) are first calculated:

$$|MSS|_{min|avg}(j) = \frac{1}{n} * \sum_{i=1}^n |MSS|_{min}(\Sigma_{tst}(i, j)|\Sigma_{trn}(i, j)), \quad (12)$$

$$|MFS_{min}|_{avg}(j) = \frac{1}{n} * \sum_{i=1}^n |MFS|_{min}(\Sigma_{trn}(i, j) | \Sigma_{trn}(i, j)). \quad (13)$$

Then, we plot the average values of $|MSS|_{min}(\Sigma_{tst} | \Sigma_{trn})$ and $|MFS|_{min}(\Sigma_{int} | \Sigma_{trn})$ against the corresponding sizes of Σ_{trn} . We call the resulting graphs as **MFS-MSS Average Curves (MMAC)** (Fig. 3).

5.2.2. MFS-MSS Matrix

Let us first introduce a new concept ‘critical section’. Within context of stide, for a splitting point Pos_i , $Size_j$ is a **critical section** $CS(i, \lambda)$ if $|MSS|_{min}(\Sigma_{tst}(i, j) | \Sigma_{trn}(i, j)) \geq \lambda$ but $|MSS|_{min}(\Sigma_{tst}(i, j-1) | \Sigma_{trn}(i, j-1)) < \lambda$. Obviously, the critical section is indispensable to provide stide detectors with the detection performance λ . Other than the critical section $CS(i, \lambda)$, the remaining part of the normal dataset can be discarded as it has negligible effect on the stide detection efficiency.

The MFS-MSS Matrix (MMM) is defined in order to help identify the critical sections in the normal dataset with respect to the predefined detection performance λ . In the matrix, the columns (the horizontal axis) are defined by the splitting sizes of the training dataset $\{Size_1, Size_2, \dots, Size_m\}$, and the rows (the vertical axis) are defined by the splitting points of the training dataset $\{Pos_1, Pos_2, \dots, Pos_n\}$ (as in Fig. 4). According to our proposed formal framework (especially from Eq. (10) and Theorem 4.17), an entry $MMM(i, j)$ in an MMM matrix will be labeled as ‘efficient’ if $|MSS|_{min}(\Sigma_{tst}(i, j) | \Sigma_{trn}(i, j)) \geq \lambda$, otherwise, it is labeled as ‘inefficient’. Furthermore, for every specific pair of Pos_i and $Size_j$, if $MMM(i, j)$ is inefficient but $MMM(i, j+1)$ is efficient, the transition from the inefficient entry $MMM(i, j)$ to the efficient entry $MMM(i, j+1)$ is named as an **efficiency transition** in the MMM Matrix. From the efficiency transition, it can be concluded that the section in Σ_{nml} from Pos_i to $(Pos_i + Size_{j+1}) \% |\Sigma_{nml}|^4$ is critical for building efficient stide detectors, i.e., it is a **critical section** $CS(i, \lambda)$.

After identifying the critical sections for all splitting points, we choose the **most compact critical section** $MCCS(\lambda) (= CS(i, \lambda))$ in the normal dataset as the training dataset for stide. As its name implies, for any other critical section $CS(k, \lambda)$ ($i \neq k$), $|MCCS(\lambda)| \leq |CS(k, \lambda)|$. Since the redundant parts in the normal dataset can be trimmed by using $MCCS(\lambda)$, the training time for the stide detectors can be substantially reduced without sacrificing the detection performance. As an added benefit, the size of $MCCS(\lambda)$ in the normal dataset provides an intuitive measure of the complexity of a process. This is because, intuitively, with respect to the same detection performance, the more complex the process is, the larger $MCCS(\lambda)$ is. Furthermore, this technique for dataset trimming can be utilized in other domains as well, such as information retrieval and computer forensic.

⁴ It is done in a wrap-round fashion like the normal dataset splitting policy in the same application.

5.2.2.1. Effect of the trimming scheme. As mentioned earlier, in order to be valid, any trimming of the training dataset must not lead to any loss in the efficiency of stide detectors. That our trimming procedure indeed satisfies the criterion is shown by the Theorem 5.1.

Theorem 5.1. Let Σ_{trn}^{cs} denote the critical section, and Σ_{tst}^{cs} the remaining part in the normal dataset. Thus, $\Sigma_{trn}^{cs} \odot \Sigma_{tst}^{cs} = \Sigma_{nml}$. The future normal dataset is denoted as Σ_{new} . Then, for all (known and unknown) intrusions with $MFS(\Sigma_{int} | \Sigma_{nml} \odot \Sigma_{new}) \leq \lambda$,

$$\begin{aligned} |MSS|_{min}(\Sigma_{new} | \Sigma_{nml}) &\geq |MFS|_{min}(\Sigma_{int} | \Sigma_{nml} \odot \Sigma_{new}) \\ \Rightarrow |MSS|_{min}(\Sigma_{tst}^{cs} \odot \Sigma_{new} | \Sigma_{trn}^{cs}) &\geq |MFS|_{min}(\Sigma_{int} | \Sigma_{nml} \odot \Sigma_{new}). \end{aligned}$$

Proof 5.2. In the trimming scheme, we assume that $|MSS|_{min}(\Sigma_{tst}^{cs} | \Sigma_{trn}^{cs}) = \lambda (>0)$, i.e. the detection performance of the stide detector built by the critical section is stable at λ to detect intrusions that cause MSSs smaller than λ . From its definition, $|MSS|_{min}(\Sigma_{tst}^{cs} \odot \Sigma_{new} | \Sigma_{trn}^{cs})$ is affected by the foreign sequence(s) $S (S \in MFS_{min}(\Sigma_{tst}^{cs} \odot \Sigma_{new} | \Sigma_{trn}^{cs}))$ under the following two scenarios:

Case 1. $S \in SS(\Sigma_{tst}^{cs})$;

$$\begin{aligned} S &\in SS(\Sigma_{tst}^{cs}), S \notin SS(\Sigma_{trn}^{cs}) \\ \Rightarrow |S| &= \lambda + 1 \\ \Rightarrow |MSS|_{min}(\Sigma_{tst}^{cs} \odot \Sigma_{new} | \Sigma_{trn}^{cs}) &= \lambda \\ \Rightarrow |MSS|_{min}(\Sigma_{tst}^{cs} \odot \Sigma_{new} | \Sigma_{trn}^{cs}) &\geq |MFS|_{min}(\Sigma_{int} | \Sigma_{nml} \odot \Sigma_{new}). \end{aligned}$$

Case 2. $S \notin SS(\Sigma_{tst}^{cs})$;

$$\begin{aligned} S &\notin SS(\Sigma_{tst}^{cs}), S \notin SS(\Sigma_{trn}^{cs}) \\ \Rightarrow S &\subset \Sigma_{new}, S \not\subset \Sigma_{nml} \\ \Rightarrow |MSS|_{min}(\Sigma_{new} | \Sigma_{nml}) &= |S| - 1 \\ \Rightarrow |MSS|_{min}(\Sigma_{new} | \Sigma_{nml}) &= |MSS|_{min}(\Sigma_{tst}^{cs} \odot \Sigma_{new} | \Sigma_{trn}^{cs}) \\ \Rightarrow |MSS|_{min}(\Sigma_{tst}^{cs} \odot \Sigma_{new} | \Sigma_{trn}^{cs}) &\geq |MFS|_{min}(\Sigma_{int} | \Sigma_{nml} \odot \Sigma_{new}) \end{aligned}$$

Based on the results under these two scenarios, the theorem is proved.

5.3. Experimental evaluations

The splitting procedure in our application works as follows. The length of the training dataset $Size_j$ is varied from 1% to 99% of the normal dataset, with a step of 7%, and the remaining portion of the normal dataset is designated as the test dataset. The splitting position Pos_i is also varied dynamically from 1% to 99% of the normal dataset, using wrap around, with a step of 7%. Thus, $m = n = 15$. The maximum length for MSSs in any test dataset is kept fixed at $N = 25$ (as all the MSSs and MFSs obtained are well within this limit).

In our experiments, the following aspects of the framework will be evaluated:

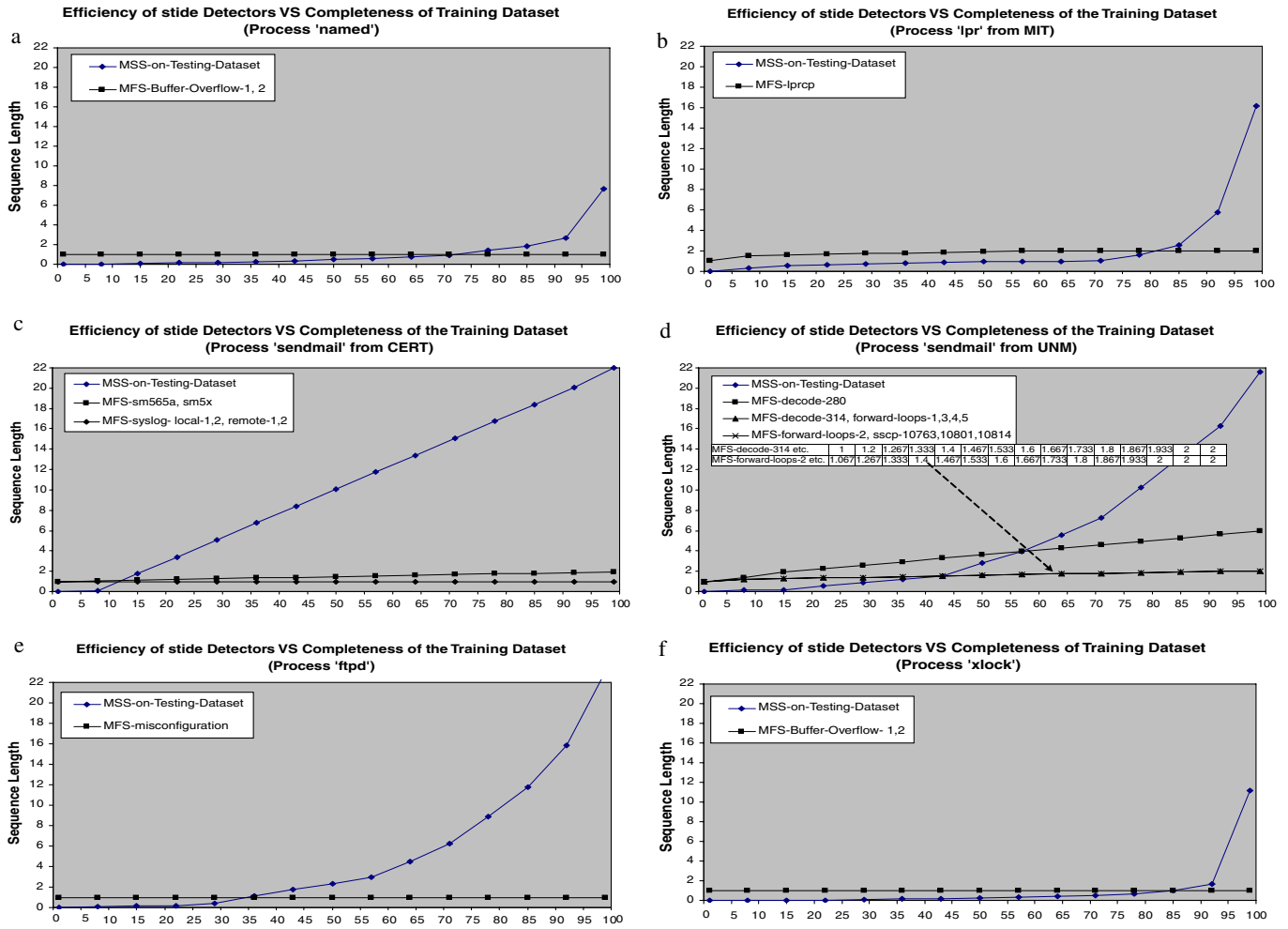


Fig. 3. MFS-MSS average curves for different processes. (a) Process 'named'. (b) Process 'lpr' from MIT. (c) Process 'sendmail' from CERT. (d) Process 'sendmail' from UNM. (e) Process 'ftpd'. (f) Process 'xlock'.

- (A) The influence of the completeness of the training dataset on the MFSs in the intrusive dataset;
- (B) The influence of the completeness of the training dataset on the MSSs in the test dataset;
- (C) The effectiveness of the trimming procedure, and the related graphical tools.

5.3.1. Evaluating the completeness of the training dataset

In Fig. 3, the MFS-MSS average curves for processes are illustrated.⁵ From these curves, the varying sensitivity of the stide detectors to the completeness of the training dataset is obvious. For processes 'named', 'xlock' and 'lpr' from MIT, the efficiency of stide detectors is quite sensitive to the completeness of the training dataset. For process 'sendmail'

from CERT, the efficient stide detectors can be obtained even with a small size of the training dataset. At the same time, the minimum foreign sequences of the intrusive dataset are not affected that much by the completeness of the training dataset since $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$ is small (*less than 2*) for 'named', 'lpr' and 'sendmail' from CERT (Eq. (10)). However, if the MFS(s) of an intrusion is larger than 2, such as 'decode-280', the influence of the minimum common false positive sequences in the intrusive dataset can be observed clearly as the completeness of the normal dataset increases. In addition, the answer to the 'Why 6?' [23] question is provided explicitly by Fig. 3d.

General speaking, the degree of sensitivity of the stide detectors to the completeness of the training dataset may be influenced by the complexity of the processes and/or the audit trails collection tools. If the function of a process is simple, the complete training dataset is easy to collect, and the detection efficiency is more influenced by the intrusion characteristics $|MFS|_{min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$ (Theorem 4.15). Otherwise, the completeness of the training dataset is hard to be guaranteed, and the stide

⁵ For clarity, in this figure, we have grouped the intrusions which have the same MFS sequences with the increase of the completeness of the training dataset. For the same reason, some MFS sequences will be organized in a table if they are too near to be distinguishable from their curves, such as the table in Fig. 3d.

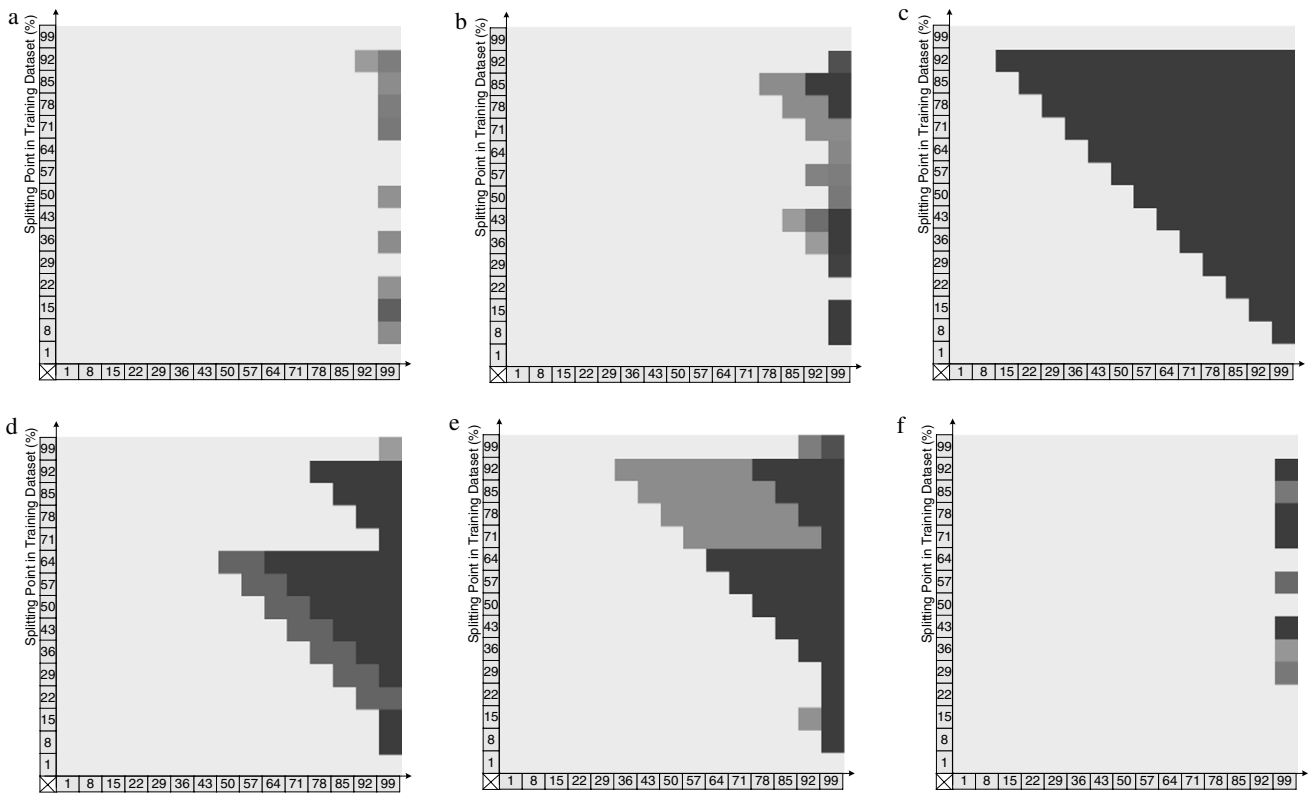


Fig. 4. MFS-MSS Matrix for different processes ($\lambda = 6$). (a) Process ‘named’. (b) Process ‘lpr’. (c) Process ‘sendmail’ from CERT. (d) Process ‘sendmail’ from UNM. (e) Process ‘ftpd’. (f) Process ‘xlock’.

detector is more sensitive to the completeness of the training dataset.

In summary, stide is very efficient to detect intrusions into a process (i.e., a computing resource) with simple function, but its efficiency will be deteriorated when detecting a complex process. For instance, stide is not appropriate to detect the intrusions in Internet since the traffic behaviors in Internet are very dynamic, and even evolved with time.

5.3.2. Identifying critical sections using MMM

In the MMM matrix, the efficiency of every entry is indicated by its darkness, which is defined by the value $|MSS|_{\min}(\Sigma_{tsi}(i, j)|\Sigma_{trm}(i, j)) - \lambda$. As in Fig. 4, the darker elements of the matrix indicate the *efficient entries*, and the lighter elements of the matrix indicate the *inefficient entries*. Therefore, based on Theorem 4.17, the darker the entry in the MMM matrix is, the more possibility to train efficient stide detectors from the training dataset determined by the entry (i, j) . Furthermore, for every splitting point, the efficiency transition is clearly visible as it is the transition from a lighter entry to the *first* darker one.

In our experiments, we let $\lambda = 6$. From these efficiency transitions in the MMM matrices (Fig. 4), the critical sections in the normal dataset for any process can be identified easily for every splitting point. For example, for the process ‘lpr’, at the splitting point $Pos_{12} = 78\%$,

$CS(12,6) = [78\%, 100\%] \cup [0, 63\%]$. Finally, the most compact critical section is gotten for every process (e.g. for ‘lpr’, $MCCS(6) = [85\%, 100\%] \cup [0, 63\%]$). In our experiments, the MCCSs for various processes in the normal datasets are shown in Table 3. Obviously, the beginning part $[0, 7\%]$ and the end part $[92\%, 100\%]$ are included in all these most compact critical sections. As discussed in [16], the beginning and end transactions of a process are critical in building the normal behavior model, and thus affect the stide efficiency.

From the critical section set for every normal dataset in MMM matrix, the sensitivity of the efficiency of stide detectors to the constitution and completeness of the training dataset can be identified more meticulously. Furthermore, from the actual size (Note, not the percentage) of $MCCS(\lambda)$ in the normal dataset of a process, we can get a rough indication of the complexity of the process. From Table 3, we can infer the following order in terms of complexity among various processes in the experimental datasets: “wu-ftp \leq sendmail from CERT \leq xlock \leq sendmail from UNM \leq lpr from MIT \leq named”.

6. Conclusions and future work

In this paper, a general framework is proposed to determine the operational limits of stide detectors. Tan and Maxion [24] in their attempt to solve the “Why six?” problem, identified the length of the minimum foreign sequence

Table 3
The most compact critical sections when $\lambda = 6$

Process	$MCCS(\lambda)(\%)$	$ MCCS(\lambda) $
'named' from UNM	$[92, 100] \cup [0, 84]$	92% – 8492126
'lpr' from MIT	$[85, 100] \cup [0, 63]$	78% – 2282517
'sendmail' from CERT	$[92, 100] \cup [0, 7]$	15% – 236412
'sendmail' from UNM	$[64, 100] \cup [0, 14]$	50% – 899882
'wu-ftp' from UNM	$[92, 100] \cup [0, 28]$	36% – 64913
'xlock' from UNM	$[71, 100] \cup [0, 70]$	99% – 335785

in the audit data as a lower bound for the length of stide detectors. Our work complements their effort by showing the effect of completeness of the normal model on stide's performance, and establishing an upper bound for the length of the detector. In addition to generalizing Tan and Maxion's results, this framework provides a formal ground for analyzing future stide-like AID detectors that are based on sequence analysis (without using frequency or probability information of sequences), by exploring the dynamics of the various factors affecting operational limits of stide, i.e. the false positives and true positives. Based on the formal framework, the foundations of several work related to stide are interpreted in a logical way.

The experiments we conducted not only validate our theoretical results, they also provide further insights by clearly showing the inter-dependencies of the various factors affecting stide's performance, i.e., the influence of completeness of the training dataset on stide efficiency is evaluated. The conclusion on the completeness evaluation is that stide is not appropriate for dynamic scenarios, such as the traffic behaviors in Internet. Then, one application is also designed to demonstrate the usefulness of our framework. It is a trimming procedure for the normal dataset, in which the redundant parts in the normal dataset are filtered out for further analysis. To achieve them, two graphical tools are designed to identify the influence and the most compact critical section: MFS-MSS Average Curve (MMAC) and MFS-MSS matrix (MMM).

From the MMAC curves, the influence of the completeness of the training dataset on the MSSs in the test dataset and the MFSs in the intrusive dataset are analyzed. The existence of the minimum common false positive sequences are also confirmed in the MMAC curves. At a finer granularity, the MMM matrix is utilized to find the most compact critical section within the normal dataset for a specific detection performance λ . The MMAC curves and the MMM matrix also provide an intuitive indication of the complexity of the corresponding process.

In this framework, the questions related to the 'Why 6?' problem can be answered clearly, such as the question in [24], 'to what extent can we establish a link between detectable anomalies and intrusive behaviors?', and the answer lies in Theorem 4.17. After analyzing the influence of the completeness of the training dataset of a process on

the efficiency of the stide detectors, we can determine whether stide is appropriate for detecting any intrusion into that process.

Limitations of the framework. However, while using the proposed framework, we should bear in mind certain limitations of the framework. These are briefly stated below.

- (1) Like any AID technique, the framework is based on the assumption that any anomaly in the intrusive dataset is an indication of an intrusion into the resource. Even though the assumption is reasonable practically under most circumstances, it is possible that a non-malicious access that deviates from the normal behavior will be detected as an intrusion. On the other hand, if an intrusion can successfully mimic normal behaviors, then no AID techniques can detect such an intrusion [25].
- (2) The framework specifically deals with stide-like AID techniques which assumes that intrusions are manifested in the sequences of system calls without using the sequence frequency/probability information and tries to detect them by a systematic analysis of such sequences. Therefore, it may not be appropriate to apply it to all possible intrusions or detection techniques [28,25].

In our future work, the framework will be further evaluated by the datasets under different environments, e.g. the networks and the windows platform. Then, the definitions for effective, complete and efficient anomaly-based intrusion detectors will be generalized to other sequence-based AID techniques.

Acknowledgements

We are grateful to Prof. Forrest and her research group at the University of New Mexico, for their scientific generosity, since the authors downloaded their datasets and their documents from [7].

Appendix A. Proofs of some Theorems

A.1. Proof of Theorem 2.4

Proof A.1. Assume that $S \in MFS_{min}(\Sigma_{igt}|\Sigma_{ref})$. Thus, $|S| = |MFS|_{min}(\Sigma_{igt}|\Sigma_{ref})$. From the definition of MFS $(\Sigma_{igt}|\Sigma_{ref})$, $S \in FRGN(\Sigma_{igt}|\Sigma_{ref})$, and there exists an 1-order subsequence $S' \in SELF(\Sigma_{igt}|\Sigma_{ref})$ at least considering that $FRGN(\Sigma_{igt}|\Sigma_{ref}) \cup SELF(\Sigma_{igt}|\Sigma_{ref}) = SS(\Sigma_{igt})$. Therefore, $(S' \in SELF(\Sigma_{igt}|\Sigma_{ref})) \wedge (S \in SS(\Sigma_{igt})) \wedge (S \succcurlyeq_1 S') \wedge (S \notin SELF(\Sigma_{igt}|\Sigma_{ref})) = True$, and we can conclude that $S' \in MSS(\Sigma_{igt}|\Sigma_{ref})$.

Then, using proof by contradiction to prove that $S' \in MSS_{min}(\Sigma_{igt}|\Sigma_{ref})$. If $S' \notin MSS_{min}(\Sigma_{igt}|\Sigma_{ref})$, there must be another sequence $S'' \in MSS_{min}(\Sigma_{igt}|\Sigma_{ref})$, and $|S''| < |S'|$. Following above deduction, we can determine that there is one $S''' \in MFS(\Sigma_{igt}|\Sigma_{ref})$, and $S''' \succcurlyeq_1 S''$. Considering $|S| = |S'| + 1$ and $|S''| = |S'''| + 1$, we get

$|S'''| < |S|$, which contradicts to $|S| = |MFS|_{\min}(\Sigma_{tgt}|\Sigma_{ref})$ as $S''' \in MFS(\Sigma_{tgt}|\Sigma_{ref})$.

Considering that $S \in MFS_{\min}(\Sigma_{tgt}|\Sigma_{ref})$, $S' \in MSS_{\min}(\Sigma_{tgt}|\Sigma_{ref})$, and $|S| = |S'| + 1$, the following equation is held:

$$|MSS|_{\min}(\Sigma_{tgt}|\Sigma_{ref}) = |MFS|_{\min}(\Sigma_{tgt}|\Sigma_{ref}) - 1. \quad \square$$

A.2. Proof of Theorem 4.2

Proof A.2. From the definitions, we know that:

$$|MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) = \min(\{l | l > 0; SS(\Sigma_{int}, l) - SS(\Sigma_{trn}, l) \neq \Phi\})$$

(1) If $\omega \geq |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn})$, then,

$$SS(\Sigma_{int}, \omega) - SS(\Sigma_{trn}, \omega) \neq \Phi \Rightarrow TPSS(\Sigma_{int}|\Sigma_{trn}, \omega) \neq \Phi$$

Hence, the stide detector of length ω built from Σ_{trn} is effective w.r.t Σ_{int} .

(2) If the stide detector with the length ω built by Σ_{trn} is effective w.r.t Σ_{int} , then,

$$TPSS(\Sigma_{int}|\Sigma_{trn}, \omega) \neq \Phi \Rightarrow SS(\Sigma_{int}, \omega) - SS(\Sigma_{trn}, \omega) \neq \Phi \Rightarrow \omega \geq |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) \quad \square$$

A.3. Proof of Theorem 4.5

Proof A.3. From the definitions, we know that:

$$|MSS|_{\min}(\Sigma_{ist}|\Sigma_{trn}) = \max(\{l | l \geq 0; SS(\Sigma_{ist}, l) - SS(\Sigma_{trn}, l) = \Phi\})$$

(1) If $\omega \leq |MSS|_{\min}(\Sigma_{ist}|\Sigma_{trn})$, then,

$$SS(\Sigma_{ist}, \omega) - SS(\Sigma_{trn}, \omega) = \Phi \Rightarrow FPSS(\Sigma_{ist}|\Sigma_{trn}, \omega) = \Phi$$

Hence, the stide detector with the length ω built by Σ_{trn} is complete w.r.t Σ_{ist} .

(2) If the stide detector with the length ω built by Σ_{trn} is complete w.r.t Σ_{ist} , then,

$$FPSS(\Sigma_{ist}|\Sigma_{trn}, \omega) = \Phi \Rightarrow SS(\Sigma_{ist}, \omega) - SS(\Sigma_{trn}, \omega) = \Phi \Rightarrow \omega \leq |MSS|_{\min}(\Sigma_{ist}|\Sigma_{trn}) \quad \square$$

A.4. Proof of Theorem 4.15

Proof A.4. From the MFS definition,

$$|MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) = \min(\{l | l > 0; SS(\Sigma_{int}, l) - SS(\Sigma_{trn}, l) \neq \Phi\})$$

The foreign sequence length vector for Σ_{trn} and Σ_{int} is

$$FSLV(\Sigma_{int}|\Sigma_{trn}) = \{l | l \geq 0; SS(\Sigma_{int}, l) - SS(\Sigma_{trn}, l) \neq \Phi\}$$

For $\forall l \in FSLV(\Sigma_{int}|\Sigma_{trn})$.

$$SS(\Sigma_{int}, l) - SS(\Sigma_{trn}, l) \neq \Phi$$

$$\iff \exists S(S \in (SS(\Sigma_{int}, l) - SS(\Sigma_{trn}, l)))$$

$$\iff \exists S(S \in SS(\Sigma_{int}, l) \wedge S \notin SS(\Sigma_{trn}, l))$$

$$\iff \exists S(S \in SS(\Sigma_{int}, l) \wedge S \notin SS(\Sigma_{trn}, l) \wedge (S \in SS(\Sigma_{ist}, l) \vee S \notin SS(\Sigma_{ist}, l)))$$

$$\iff \exists S(S \in SS(\Sigma_{int}, l) \wedge S \notin SS(\Sigma_{trn}, l) \wedge (S \in SS(\Sigma_{ist}, l) \vee (S \in SS(\Sigma_{int}, l) \wedge S \notin SS(\Sigma_{trn}, l) \wedge S \notin SS(\Sigma_{ist}, l))))$$

$$\iff \exists S(S \in SS(\Sigma_{int}, l) \wedge S \notin SS(\Sigma_{trn}, l) \wedge (S \in SS(\Sigma_{ist}, l) \vee (S \in SS(\Sigma_{int}, l) \wedge S \notin (SS(\Sigma_{trn}, l) \vee (SS(\Sigma_{ist}, l))))))$$

$$\iff \exists S((S \in SS(\Sigma_{int}, l) \wedge S \in (SS(\Sigma_{ist}, l) - SS(\Sigma_{trn}, l))) \vee (S \in SS(\Sigma_{int}, l) \wedge S \notin SS(\Sigma_{trn} \odot \Sigma_{ist}, l)))$$

$$\iff \exists S((S \in SS(\Sigma_{int}, l) \wedge S \in FPSS(\Sigma_{ist}|\Sigma_{trn}, l)) \vee (S \in SS(\Sigma_{int}, l) \wedge S \notin SS(\Sigma_{trn} \odot \Sigma_{ist}, l)))$$

$$\iff \exists S(S \in (SS(\Sigma_{int}, l) \cap FPSS(\Sigma_{ist}|\Sigma_{trn}, l)) \vee S \in (SS(\Sigma_{int}, l) - SS(\Sigma_{trn} \odot \Sigma_{ist}, l)))$$

$$\iff \exists S((S \in SS(\Sigma_{int}, l) \cap FPSS(\Sigma_{ist}|\Sigma_{trn}, l)) \cup (SS(\Sigma_{int}, l) - SS(\Sigma_{trn} \odot \Sigma_{ist}, l)))$$

$$\iff SS(\Sigma_{int}, l) \cap FPSS(\Sigma_{ist}|\Sigma_{trn}, l) \neq \Phi \vee SS(\Sigma_{int}, l) - SS(\Sigma_{trn} \odot \Sigma_{ist}, l) \neq \Phi$$

Hence,

$$FSLV(\Sigma_{int}|\Sigma_{trn}) = \{l | l \geq 0; SS(\Sigma_{int}, l) \cap FPSS(\Sigma_{ist}|\Sigma_{trn}, l) \neq \Phi\} \cup \{l | l \geq 0; SS(\Sigma_{int}, l) - SS(\Sigma_{trn} \odot \Sigma_{ist}, l) \neq \Phi\}$$

Finally,

$$\begin{aligned} & |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) \\ &= \min(FSLV(\Sigma_{int}|\Sigma_{trn})) \\ &= \min(\{l | SS(\Sigma_{int}, l) \cap FPSS(\Sigma_{ist}|\Sigma_{trn}, l) \neq \Phi\} \\ &\quad \cup \{l | SS(\Sigma_{int}, l) - SS(\Sigma_{trn} \odot \Sigma_{ist}, l) \neq \Phi\}) \\ &= \min(\min(\{l | SS(\Sigma_{int}, l) \cap FPSS(\Sigma_{ist}|\Sigma_{trn}, l) \neq \Phi\}), \\ &\quad \min(\{l | SS(\Sigma_{int}, l) - SS(\Sigma_{trn} \odot \Sigma_{ist}, l) \neq \Phi\})) \\ &= \min(|CFPS|_{\min}(\Sigma_{int}, \Sigma_{ist}|\Sigma_{trn}), |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{ist})) \quad \square \end{aligned}$$

A.5. Proof of Theorem 4.17

Proof A.5. If $|MSS|_{\min}(\Sigma_{ist}|\Sigma_{trn}) \geq 1$,

$$\forall l(1 \leq l \leq |MSS|_{\min}(\Sigma_{ist}|\Sigma_{trn}), FPSS(\Sigma_{ist}|\Sigma_{trn}, l) = \Phi) \quad (A.1)$$

Furthermore, according to the definition of CFPS,

$$|CFPS|_{\min}(\Sigma_{int}, \Sigma_{ist}|\Sigma_{trn}) > |MSS|_{\min}(\Sigma_{ist}|\Sigma_{trn}) \quad (A.2)$$

(\Leftarrow) If $|MSS|_{\min}(\Sigma_{ist}|\Sigma_{trn}) \geq |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{ist})$, there exist efficient stide detectors for datasets Σ_{trn} , Σ_{ist} and Σ_{int} .

$$\begin{aligned}
& |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) \geq |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) \\
& \Rightarrow |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) \geq 1 \\
& \Rightarrow |CFPS|_{\min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) > |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) \\
& \Rightarrow |CFPS|_{\min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) > |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) \\
& \Rightarrow |MFS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) = |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) \\
& \Rightarrow |MFS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) \leq |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn})
\end{aligned}$$

From Theorem 4.9, there exist efficient stide detectors under this scenario.

(\Rightarrow) If there exist efficient stide detectors for Σ_{trn} , Σ_{tst} and Σ_{int} , $|MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) \geq |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$.

To apply the proof by contradiction, let us assume $|MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) < |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$.

(§1) If $|MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) = 0$.

From the MFS definition, $|MFS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) \geq 1 > |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn})$. Based on Theorem 4.9, there does not exist efficient stide detectors, and that is contradict with our statement. So, $|MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) < |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$ is not correct.

(§2) If $|MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) \geq 1$,

(2.a) If $|CFPS|_{\min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) \geq |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$.

From Eq. (10), $|MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) = |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$. Furthermore, we assume $|MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst}) > |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn})$, therefore

$$|MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) > |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn})$$

(2.b) If $|CFPS|_{\min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) < |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$.

From Eq. (10), $|MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) = |CFPS|_{\min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn})$. and Eq. (A.2).

$$|CFPS|_{\min}(\Sigma_{int}, \Sigma_{tst}|\Sigma_{trn}) > |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn})$$

$$\Rightarrow |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn}) > |MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}).$$

From (2.a), (2.b) and Theorem 4.9, there are no efficient stide detectors, and that is contradict to the statement. Therefore, $|MSS|_{\min}(\Sigma_{tst}|\Sigma_{trn}) < |MFS|_{\min}(\Sigma_{int}|\Sigma_{trn} \odot \Sigma_{tst})$ is not correct.

Based on (1) and (2), the theorem is proved. \square

References

- [1] J.P. Anderson, Computer Security Threat Monitoring and Surveillance. Technical report, James P Anderson Co., Fort Washington, Pennsylvania, April 1980.
- [2] S. Axelsson, The base-rate fallacy and its implications for the difficulty of intrusion detection, in: Proceedings of the Sixth ACM Conference on Computer and Communications Security, 1999, pp. 1–7.
- [3] S.N. Chari, P. Cheng, BlueBox: a policy-driven, host-based intrusion detection system, ACM Transaction on Information and System Security 6 (2) (2003) 173–200.
- [4] K. Calvin, F.M. Ruschitzka, L. Karl, Execution monitoring of security-critical programs in distributed systems: a specification-based approach, in: 1997 IEEE Symposium on Security and Privacy, Oakland, CA, May 04–07, 1997.
- [5] H. Debar, M. Dacier, A. Wespi, Reference audit information generation for intrusion detection systems, in: Proceedings of IfipSec 98, Vienna, Austria, August 1998.
- [6] S. Forrest, S.A. Hofmeyr, A. Somayaji, T.A. Longstaff, A sense of self for Unix processes, in: Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press, 1996, pp. 120–128.
- [7] S. Forrest, University of new mexico, datasets for stide. <http://www.cs.unm.edu/immsec>, 1994.
- [8] S. Forrest, A.S. Perelson, L. Allen, R. Cherukuri, Self-nonsel self discriminant in a computer, in: Proceedings of IEEE Symposium on Research in Security and Privacy, Oakland, CA, May 16–18, 1994, pp. 202–212.
- [9] S.A. Hofmeyr, S. Forrest, Architecture for an artificial immune system, Evolutionary Computation 8 (4) (2000) 443–473.
- [10] S.A. Hofmeyr, S. Forrest, A. Somayaji, Intrusion detection using sequences of system calls, Journal of Computer Security 6 (3) (1998) 151–180.
- [11] S. Jha, K.M.C. Tan, R.A. Maxion, Markov chains, classifiers, and intrusion detection, in: Proceedings of the Fourteenth IEEE Computer Security Foundations Workshop, 2001, pp. 206–219.
- [12] H. Javits, A. Valdes, The NIDES Statistical Component: Description and Justification. SRI Annual Report A010, SRI International, Computer Science Laboratory, March 1993.
- [13] W. Ju, Y. Vardi, A hybrid high-order markov chain model for computer intrusion detection, Journal of Computational and Graphical Statistics 10 (2) (2001) 277–295.
- [14] R.A. Kemmerer, G. Vigna, Intrusion detection: a brief history and overview. IEEE Computer, 35(4):suppl27–suppl30, April 2002.
- [15] T.D. Lane, C.E. Brodley, Sequence matching and learning in anomaly detection for computer security, in: Fawcett, Haimowitz, Provost, Stolfo (Eds.), AI Approaches to Fraud Detection and Risk Management, AAAI Press, 1997, pp. 43–49.
- [16] Zhuowei Li and Amitabha Das. Visualizing and identifying intrusion context from system calls trace, in: Proceedings of 20th Annual Computer Security Applications Conference, December 2004.
- [17] W. Lee, S.J. Stolfo, A framework for constructing features and models for intrusion detection systems, ACM Transactions on Information and System Security 3 (4) (2000) 227–261.
- [18] W. Lee, D. Xiang, Information-theoretic measures for anomaly detection, in: Proceedings of the 2001 IEEE Symposium on Security and Privacy, 2001, pp. 130–143.
- [19] M.V. Mahoney, P.K. Chan, Learning nonstationary models of normal network traffic for detecting novel attacks, in: SIGKDD 2002, July 23–26, 2002.
- [20] P.G. Neumann, P.A. Porras, Experience with EMERALD to date, in: Proceedings of First USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, California, April 11–12, 1999, pp. 73–80.
- [21] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, S. Zhou. Specification-based anomaly detection: a new approach for detecting network intrusions, in: Proceedings of the Ninth ACM conference on Computer and communications security, 2002, pp. 265–274.
- [22] M. Stillerman, C. Marceau, M. Stillman, Intrusion detection for distributed applications, Communications of the ACM 42 (7) (1999) 62–69.
- [23] K.M.C. Tan, R.A. Maxion, Why 6? defining the operational limits of stide, an anomaly-based intrusion detector, in: Proceedings of 2002 IEEE Symposium on Security and Privacy, Berkeley, California, May 12–15, 2002, pp. 173–186.
- [24] K.M.C. Tan, R.A. Maxion, Determining the operational limits of an anomaly-based intrusion detector, IEEE Journal on selected areas in communications 21 (1) (2003) 96–110.
- [25] K.M.C. Tan, J. Mchugh, K.S. Killourhy. Hiding intrusions: from the abnormal to the normal and beyond, in: Proceedings of Information Hiding 2002, 2002, pp. 1–17.

- [26] A. Wespi, M. Dacier, H. Debar, Intrusion detection using variable-length audit trail patterns, in: Proceedings of the Third International Workshop on the Recent Advances in Intrusion Detection, LNCS 1907, 2000, pp. 110–129.
- [27] C. Warrender, S. Forrest, B.A. Pearlmutter, Detecting intrusions using system calls: alternative data models, in: IEEE Symposium on Security and Privacy, 1999, pp. 133–145.
- [28] David Wagner, Paolo Soto, Mimicry attacks on host-based intrusion detection systems, in: Proceedings of the Ninth ACM conference on Computer and Communications Security, 2002, pp. 255–264.
- [29] N. Ye. A markov chain model of temporal behavior for anomaly detection, in: Proceedings of the 2000 IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop 2000, 2000, pp. 171–174.

Author's personal copy