# Impact of IT Monoculture on Behavioral End Host Intrusion Detection

Dhiman Barman
UC Riverside, Juniper

Jaideep Chandrashekar
Intel Labs Berkeley

Nina Taft
Intel Labs Berkeley

Michalis Faloutsos *
UC Riverside

Ling Huang
Intel Labs Berkeley

Frederic Giroire †
MASCOTTE, I3S (CNRS, UNS) - INRIA, France

## ABSTRACT

In this paper, we study the impact of today's IT policies, defined based upon a monoculture approach, on the performance of end-host anomaly detectors. This approach leads to the uniform configuration of Host intrusion detection systems (HIDS) across all hosts in an enterprise networks. We assess the performance impact this policy has from the individual's point of view by analyzing network traces collected from 350 enterprise users. We uncover a great deal of diversity in the user population in terms of the "tail" behavior, i.e., the component which matters for anomaly detection systems. We demonstrate that the monoculture approach to HIDS configuration results in users that experience wildly different false positive and false negatives rates. We then introduce new policies, based upon leveraging this diversity and show that not only do they dramatically improve performance for the vast majority of users, but they also reduce the number of false positives arriving in centralized IT operation centers, and can reduce attack strength.

## Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection; C.2.0 [**Computer-Communication Networks**]: General

## General Terms

Management, Measurement, Security, Performance

## Keywords

Host Intrusion Detection, Anomaly Detection, User Profile, Measurement, Enterprise Management

*Research was supported by NSF 0831530 and NSF 0832069 and an Intel gift.

†Research was supported by European project IST FET AEOLUS and the ANR PROJECTS SPREADS and DIMAGREEN.


*WREN'09,* August 21, 2009, Barcelona, Spain.


## 1. INTRODUCTION

Host Intrusion Detection Systems (HIDS) have become ubiquitous in enterprise networks since all laptops and desktops issued to employees come preconfigured with a HID system on it. These end host applications thus form a key part of the overall enterprise security infrastructure, as they constitute the last line of defense. These systems are typically configured to interact with centralized IT management; for example, end hosts receive periodic updates (new signatures) and often batch alerts that are sent periodically to IT. Recent trends in commercial HIDS, called *behavioral* HIDS, have led to systems that perform not only signature detection (not studied here as they cannot detect previously unknown attacks), but also anomaly detection based upon user profiling. However, today's behavioral HIDS are still fairly primitive and can monitor just a few features. Recent hardware capabilities, such as those introduced by Intel's AMT [2], allows for in-hardware monitoring of a limited number of traffic features. It is not hard to imagine that in the future such hardware will be able to track large numbers of features simultaneously. As behavioral HIDS evolve, they will thus incorporate a larger and larger set of monitored features. Despite this impending development, there is little conventional wisdom about how to configure the thresholds that these detectors use. As the same time, the debate about the monoculture approach of IT management has resurfaced [7]. "It is believed that a collection of identical computing platforms is easier, hence cheaper, to manage because making one set of configuration decisions suffices for all." Some [7] argue that deploying a monoculture could be a good defense if the configurations used are well tested and well understood. Others claim that monocultures benefit attackers because if they can successfully control one host, they can do it repeatedly on many hosts. Those that believe the latter have suggested introducing artificial diversity to confuse attackers ( [7, 9, 26, 27]).

In this work, we study the impact of IT monoculture policies on performance in the face of these two trends, the increasing use of behavioral-based approaches to HIDS, and no coherent methodology for threshold selection of such systems. In particular we challenge the common enterprise practice of configuring HID systems on laptops for an employee population all the same way. First we study the impact of such an approach on individual users. We collect traffic traces from over 350 end-hosts in a large enterprise network and demonstrate that there is tremendous natural diversity in user fringe behavior. "User fringe behavior" loosely refers to the outlier behavior of a user in terms of the features typically monitored by anomaly detectors. While it is generally believed that the user population is diverse, our study is the first to quantify diversity in the tail of feature distributions. We show that a monoculture policy to HIDS configuration leads to a situation in which individual users experience dramatically different performance from one

to another in terms of false positives and false negatives. This is because such policies implicitly ignore the underlying user diversity.

We then propose and assess diversity configuration policies, and show that the vast majority of users experience far superior performance under such policies. However we ask under what conditions do the benefits of diversity policies surface. We show that the gains of diversity over monoculture increase as the relative importance of minimizing false negatives to minimizing false positives grows. In other words, if minimizing false positives is an enterprise's only goal, then diversity policies bring little gain. However as minimizing missed detections increases in importance, then the benefits of diversity surface more and more. We illustrate that our diversity policy can also limit the strength of attacks thereby reducing attacker effectiveness. We illustrate this even for a resourceful attacker that can observe end user behavior.

Several system administrators we polled favor the monoculture approach for two reasons; they claim that it is easier to check compliance for a large pool of employees when homogenous configurations are used, and they say that it is hard to interpret alarms if they fire under seemingly different conditions. To strike a balance between a monoculture approach and that of a fully diverse approach, we propose a partial diversity policy in which users are put into a small number of groups. Configurations across groups are different, but users within each group are given the same configuration. Our preliminary study on partial diversity indicates that it is possible with a small number of groups to achieve most of the benefits of a fully diverse approach.

## 2. RELATED WORK

With encrypted traffic in a network growing, it becomes difficult for in-network analysis [25], calling for more research on HIDS. HIDS typically has two components, signature detection and anomaly detection. Signature detection plays an important role, however, it is not useful in detecting previously unknown attacks. The anomaly detection component tracks pre-defined features of the end-host activity, defines normal behavior and then raises alerts when abnormal behavior are observed. A major class of anomaly detectors have been designed for tracking program execution; for such detectors alerts are raised when rules on expected program behavior are violated [11, 13, 17, 25]. In another class, anomaly detectors build statistical models of application layer or networking layer traffic [18–20]. In this paper, we use the example of statistical anomaly detectors at the networking layer that are intended to uncover DDoS, scanning and bandwidth flooding attacks (often conducted through botnets).

There has been much research on developing architectures and DoS solutions that propose to mitigate DoS attacks by enabling a receiver to stop unwanted traffic [4, 12, 21, 28, 29] (among others). Capability-based systems allow a receiver to explicitly authorize the traffic it is willing to receive, and filter-based schemes enable a receiver to install network filters to block unwanted traffic. However, these designs are effective when receivers can reliably distinguish attack traffic from legitimate traffic. The HID systems we explore herein are part of the infrastructure that helps to distinguish attack traffic from legitimate traffic. More effective HIDS could complement the above systems with improved attack traffic detection. Approaches to detect and mitigate DDoS attacks due to botnets [14–16] provide defenses that observe traffic inside the enterprise infrastructure, such as at gateways, routers and so on. For example, BotSniffer [15] proposes to detect botnet traffic within a network by exploiting the spatio-temporal correlation and similarities of responses to control commands issued by botmasters on C&C Channels. HID systems differ as they lie at the edge of the network. They are resources for detailed user profile data that can be exploited to improve protection.

To increase their chances of evading detection, attackers try to hide inside normal traffic patterns by commanding zombies to increase their traffic in ways that are not too dissimilar from their regular traffic. One way to defend against mimicry attacks is to eliminate homogeneity (i.e. add diversity) in the system, so it is harder for the attacker to understand typical behaviors. There has been a large body of research propose to introduce diversity in the form of randomization at different abstraction levels (in memory locations, compiler, program execution) [8, 26]. However, the concept of exploiting diversity in HIDS-based anomaly detection systems has not been well explored [22]. Our study illustrates that applying diversity at the IT policy level of HIDS configuration can work to reduce the effectiveness of attacks, (such as limiting DoS attack strength), and improve the performance of individual hosts.

## 3. PROBLEM STATEMENT

HID systems concurrently monitor a number of traffic features, each of which is compared against an anomaly detector threshold. When a threshold is exceeded, alerts are generated and periodically sent to a central console. The particular value of the threshold determines the number of false positives (FP), which are benign alerts, and the false negatives (FN), i.e., the number of instances where there was an anomaly present without a corresponding alarm (a missed detection). We do not comment on which traffic features should be monitored; nor do we promote particular threshold values for the anomaly detectors. Instead, our goal is to illustrate the ramifications of enterprize policies on individual users when the underlying population is very diverse. As the malware threats evolve, so will the methods used to detect them; thus we expect that the particular features tracked for anomalies, and their associated thresholds will change over time.

We studied features that have been suggested in the literature [6, 24], or else are capabilities of specific commercial products [1, 3, 10]. A list of the features experimented with are enumerated in Table 1. The table indicates the type of anomaly intended to be detected by the particular feature, as well as some commercial products that incorporate it. The list of features we study is representative, but in no way exhaustive. A common property of these features is that they are all additive.

| *Feature* | *Anomaly* | *Product* |
|---|---|---|
| num-DNS-connections | Botnet C&C | Damballa |
| num-TCP-connections | scans, DDoS | Cisco CSA |
| num-TCP-SYN | scans, DDoS | BRO, CSA |
| num-HTTP-connections | Clickfraud, DDoS | BRO, BlackIce |
| num-distinct-connections | scans | BRO |
| num-UDP-connections | scans, DDoS | Cisco CSA |

**Table 1: Features Used in Our Study.**

We assume that each host in an enterprise can be potentially infected and recruited into a botnet, and the *botmaster* can potentially use all such hosts to stage a DDoS attack, send out spam, or any of the other activities hosted on botnets. We use $g_i^j$ to be the random variable representing the value of the $j^{th}$ traffic feature on end-host $i$ (i.e., one of the features described in Table 1). The botmaster can instruct the recruited hosts to send out *additional* traffic to other destination, which additively increases the feature being tracked. Let $b_i^j$ be the increase in traffic feature $j$ corresponding to the malicious traffic the botmaster inserts on user $i$'s machine. We consider

two types of attackers. One is naive and does not have any knowledge of the traffic pattern of the endhost; this attacker injects arbitrary amounts of traffic into the user's traffic. In our second model, we assume a resourceful attacker that has inserted monitoring code on the zombie's machine and can thus compute histograms of the user's behavior himself. This strong threat model captures an attacker with a great deal of knowledge about the user. We do assume that the malicious traffic is additive in the tracked feature. Thus the traffic seen by the anomaly detector tracking feature $j$ on host $i$ is given by the random variable $g_i^j + b_i^j$.

The policy chosen by the IT operator assigns a threshold $T_i^j$ to end-host $i$ for feature $j$. If at any time $g_i^j + b_i^j > T_i^j$, an alarm is raised. The false negative rate is thus defined as the probability of a missed detection, $FN_i = P(g_i^j + b_i^j < T_i^j)$. Similarly, the false alarm rate is defined as $FP_i = P(g_i^j > T_i^j)$. The problem at hand is to define good policies for selecting the $T_i^j$ values across all features and across all users such that each user experiences good performance in terms of their own $(\langle FN_i, FP_i \rangle)$. Although often not explicitly stated, enterprise IT departments do hope that all their users will experience roughly similar performance. In theory, they do not want some users to be more vulnerable than others; that is the reason for the typically strong enforcement of security policies across *all* employees.

# 4. POLICIES

We model an enterprise policy to configure the anomaly detectors in the HIDS systems as having two components, (i) a *threshold selection* heuristic, and (ii) a *grouping method*. The grouping method specifies whether each host is treated individually (i.e., number of groups equals number of hosts), whether all hosts are treated the same (i.e., a single group), or whether the hosts are organized into a small number of groups, such that all hosts within a group receive the same configuration.

Various heuristics can be applied for selecting particular threshold values. We examined multiple threshold heuristics, including percentile detectors, those based on optimizing the F-measure[1], cases when outliers are defined as the mean plus a few standard deviations, and some cases based on picking a threshold to optimize a utility function. A particular threshold value, impacts the tradeoff between false positives and false negatives, and thus determines the detector's operating point, i.e., $(\langle FP_i, FN_i \rangle)$. Due to lack of space, we present only the results for the percentile based detectors, as these are the most intuitive. (However, our basic findings hold across different threshold heuristics and can be found in [5].) This heuristic targets a particular false positive rate, without regard to the false negative rate. We conducted a survey of about a dozen IT personnel across 4 enterprises and 1 university. They all said that there are no standard rules of thumb as to how to balance the false positives versus the false negatives, when they have explicit control over threshold setting. They indicated that a common choice by IT operators today is to roughly target the $99^{th}$ percentile value; thus unless otherwise specified, this is the particular threshold selection heuristic we experiment with. The significance of the $99^{th}$ heuristic, and why it is popular, is that it provides a very explicit control on the rate of false positives (at most 1%, by definition). Furthermore, as described in [23] current enterprise defenses (against malware and other intrusions) today tend to attach a lot more importance to low false positive rates than to low missed detections.

The second part of our policy is the *grouping method*, which determines how the population of end-hosts is partitioned, and the number of thresholds that must be computed. We look at three grouping scenarios.

- In the **homogenous** scenario, all the users (end-hosts) are configured with exactly the same threshold value, determined by the IT operators. This mirrors the current model of operation for most IT departments. Each end-host computes its traffic probability distribution and ships it off to the central console (this is done for every traffic feature being monitored). At the centralized location, all the individual distributions are collapsed into a single global distribution from which the targeted percentiles are extracted.

- In the **full-diversity** scenario, each end-host independently determines its own threshold value, based on its own traffic observations. Thus, the threshold values are allowed to be completely different at each host. For the percentile heuristic, the host simply builds the empirical traffic distribution and extracts the particular percentile values (all done locally).

- In the **partial-diversity** setting, users are partitioned into a small number of groups, and within each individual group, the thresholds are set to the same value. This is similar to applying the homogenous scenario inside each group, but allowing the thresholds to be different across groups. This grouping method requires centralized reporting much like the homogenous solution. Again, all the data is pulled to the central console. The end hosts are partitioned into groups and a single threshold is computed per group per feature, which is then communicated back to the hosts. We describe an initial grouping heuristic used herein in the next section. The question of how to group endhosts is an interesting and potentially complex problem itself that we leave for future work.

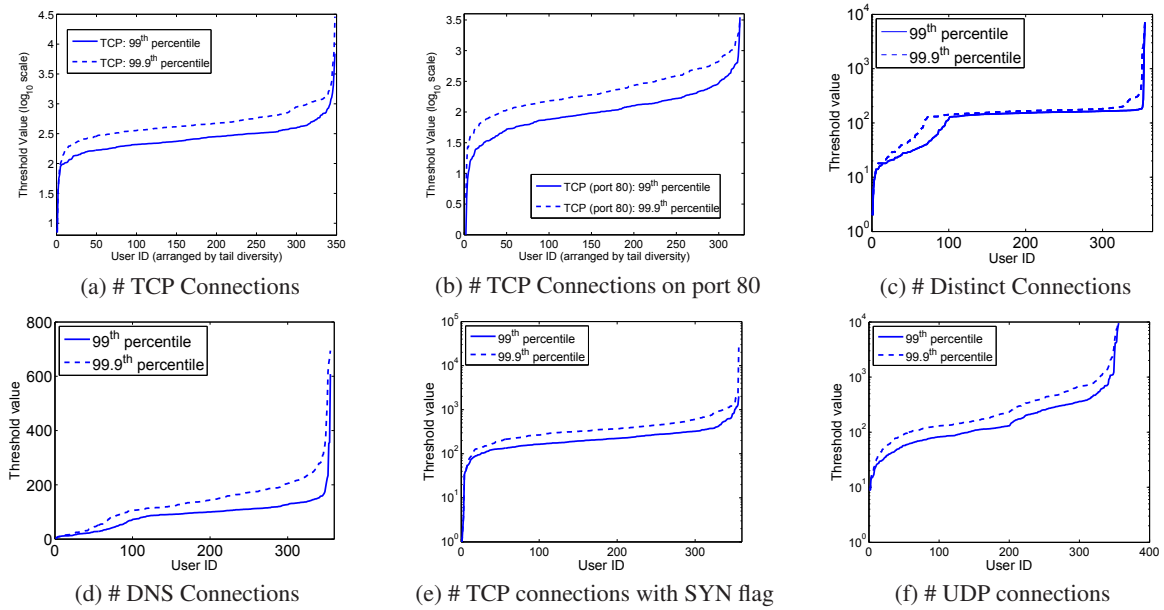
# 5. USER DIVERSITY

**Dataset:** Our data consists of network packet traces collected at 350 end-hosts (95% of them are laptops and all hosts were using Windows XP) in a large enterprise network. The traces span over 5 weeks in Q1 of 2007. Each end-host corresponds to an individual user and all users enrolled on a volunteer basis. The data collection was performed by a stand alone application (a wrapper around the `windump` tool). In addition to collecting packet headers, our collection tool watched for changes in IP address, interfaces (e.g., wired/wireless) and location. Because the collection was performed directly on the end-host, all packet activity was captured, even when the mobile laptops changed environments (home, work, different wireless interfaces, etc). This dataset captures an unusually complete view of users' behaviors; data collections that are carried out at gateways and routers do not capture user activity when they leave work or switch to another network.

**User Traffic Characteristics:** We processed the traffic traces from 350 end-hosts using the Bro tool [24] and constructed time-series for each of 6 anomaly detection features mentioned in Table 1. The counts in each time series were aggregated 5 minute (and 15 minute) bins. We present results here for the 15 minute case, noting that the conclusions hold for the shorter binning interval as well. The particular features were selected because they appear in existing anomaly detection systems ( [2, 10, 24])[2]. We treat each bin count as a sample point of the distribution $P(g_i^j)$ for the $i^{th}$ end-host. Once we obtain the distributions, $P(g_i^j)$, we compute $99^{th}$ and $99.9^{th}$ percentiles of each feature distribution as cut-off thresholds.

[1]This is defined as the harmonic mean between precision and recall, and is often suggested in the statistical literature.

[2]Features chosen from [24] were only those that are computed on a per source basis




(a) # TCP Connections (b) # TCP Connections on port 80 (c) # Distinct Connections

(d) # DNS Connections (e) # TCP connections with SYN flag (f) # UDP connections

**Figure 1: Tail diversity exists across multiple features**

Figure 1 shows the tremendous diversity in choice of thresholds that would result if this personalized policy were adopted. Interestingly, the range of diversity varies by 3 to 4 orders of magnitude for 5 of the 6 features assessed here. (The number of DNS connections varies only across two orders of magnitude.) This demonstrates in a loose sense, that the "tail", or fringe, of the user's behavior begins in very different places for different users. For example, Fig.1(a) illustrates that the beginning of these "tails" can range from 50 to 7000 for a false positive rate of 1%. We thus conclude that there is a great deal of diversity in user fringe behavior (i.e., in the outlier regions of a particular behavioral feature) and this is the part of their behavior that matters for anomaly detection.

If all hosts were to self select their cutoff thresholds this way, then the threshold would be meaningful in terms of their own behaviors, namely that all users would experience a common false positive rate. This simple point is important; we believe that IT policies should target a common user performance point rather than a common user configuration. Our study reinforces that these two goals cannot be simultaneously achieved because of the inherent user diversity.

This result indicates that some users will be more useful than others in detecting a given attack. Those with low thresholds are better suited to catchy stealthy behaviors. If one user is well suited (because of their natural behavior) to detecting an attack type A, while a second user is better suited to uncovering attack type B, then clearly these two users could collaborate to help each other. In Fig. 2 we plot the 99th percentile value for each user for two features. Each point is one user, and the 99th percentile value for TCP connections is given on the x-axis, while the value for UDP connections is given on the y-axis. We see some users at the extreme lower right of the graph. These users are "light" in terms of the maximum number of UDP connections they generate, but "heavy" in terms of the TCP traffic they generate. Such users are more likely to be useful in detecting attacks that involve abnormal UDP behavior than those involving abnormal TCP behavior. Conversely, the users in the far upper left of the plot are exactly the opposite.

As a second quick check on the potential of users to play different roles in collaborative anomaly detection, we also extracted the user identities of those with the 10 lowest thresholds, under the Diversity and Partial-Diversity policies. We did this for two features, the *number-of-UDP-connections* and the *number-of-TCP-connections*. In a loose sense, these users can be viewed as the "best" users for detecting stealthy anomalies because they can identify small anomalies since their thresholds are low. The identities of these users are stated in Table 2. The particular identity of each user is not important here. The point is that looking across the lists for these two features, for the Diversity policy, we see only 2 common users. For the Partial-Diversity policy, we only find 4 common users across the two features. This indicates that typically those users that are best for detecting attacks involving TCP connections are not the same as those that are the best for detecting attacks involving UDP connections. Figure 2 and Table 2 indicate that when user thresholds are personalized, it opens the door to new and interesting possibilities to explore defenses in which users play different roles.

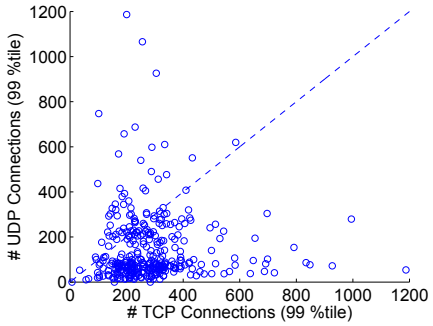



**Figure 2: Each point is one user. Comparison of fringe behavior of two features for individual users.**

**Grouping Users:** We tried to use a simple *k-means* approach to finding clusters over the user population in order to evaluate the partial diversity grouping policy. Specifically, we attempted to cluster