Anomaly Detection under Poisoning Attacks

Radhika Bhargava
Department of Computer Science
Purdue Universitye
West Lafayette, IN
bhargavr@cs.purdue.edu

Chris Clifton
Department of Computer Science and CERIAS
Purdue University
West Lafayette, IN
clifton@cs.purdue.edu

ABSTRACT

Anomaly detection has been widely applied in domains such as credit card fraud detection and intrusion detection. Whenever anomaly detection techniques are applied to security domains, we should assume that an adversary will attempt to evade the protective measures. Hence, to improve the efficacy of anomaly detection techniques, it is imperative to evaluate these techniques against adversaries. We propose a framework to assess the vulnerability of anomaly detection techniques against adversaries with the ability to poison the training dataset. We demonstrate that we can predict the expected number of attack samples an attacker needs to disguise the actual data point from a DBSCAN-based anomaly detection method. We validate our framework on the KDDCup'99 and Yahoo (S5) Anomaly Detection datasets.

CCS Concepts

•**Security and privacy** → *Intrusion/anomaly detection and malware mitigation;* •**Computing methodologies** → *Machine learning;*

Keywords

Adversarial Machine Learning, Anomaly Detection, DBSCAN, Clustering, Security Evaluation, Computer Security

1. INTRODUCTION

Anomalies are identified as anything that deviates from the normal behavior or does not conform to the common pattern. Anomaly detection attempts to identify these anomalies or outliers and is used extensively in applications like network intrusion detection, credit card fraud detection, fault detection, etc. Anomaly detection is important because the anomalous items translate into significant and actionable information in a wide variety of application domains [11]. E.g., anomalies in credit card data could signify that credit card fraud or identity-theft has taken place. Anomalies in medical diagnostic images may indicate the presence of a disease or malignant tumors. Anomalies in body sensor data can be analyzed to provide early health warnings [16], or in vibration, data could be evidence of a faulting machine [1].

Clustering, one of the anomaly detection technique, assumes that the normal data belongs to a large or dense cluster and anomalies do not belong to any cluster or belong to a sparse cluster. DBSCAN [7] is a popular density based clustering algorithm for anomaly detection and has been effectively applied in a variety of domains [9, 29, 22]. The method assumes that the normal data lies in high density regions whereas anomalies or outliers are the points that lie in low density regions. One of the advantage of the density based method is that it is good at discovering clusters of arbitrary shapes.

If we hope to use machine learning as a general tool for anomaly detection, especially for security applications like intrusion detection and fraud detection, then we can assume that whenever machine learning techniques are used to provide protection against illegal activities, adversaries will try to find a way to circumvent the techniques [21]. The challenge addressed in this paper is that in many of these applications, the anomalous entity may actively try to escape detection. Machine learning methods are built on the assumption that the training data is independent and identically distributed according to an unknown probability distribution which makes the methods vulnerable to adversarial manipulation [5, 4, 19, 28].[1]

The problem of learning in adversarial environments has recently gained increasing popularity, and relevant research has been done. Specific attacks have been devised to subvert the learning process [10, 2]. On the other hand, to the best of our knowledge, few works have explicitly addressed the issue of security evaluation related to the application of anomaly detection using clustering [20]. In this work, we propose a framework to address situations where the adversary is unable to change their own data, but *can* create fake entities to attempt to make their actual data look less like an anomaly. The goal is to help the defender estimate the risk posed by such attacks by estimating the effort required by the attacker to disguise the anomalous point under the constraint that he cannot change the anomalous point. We then evaluate how vulnerable DBSCAN-based anomaly detection is to such an attack.

ODDv5.0 at KDD August 19–23, 2018, London, UK

ISBN . . . $15.00

DOI:

[1]Note that we are using adversarial machine learning in the sense of [18]. There has been recent use of the term adversarial in the sense of using two machine learning models to train each other, e.g., [17]; this is significantly different from the issue addressed in this paper.

Novel contributions of this paper include developing an optimal attack policy by exploiting the underlying properties of the machine learning algorithm and modeling vulnerability to this attack. We aim to answer the following questions:

1. Can an adversary leverage knowledge about the anomaly detection technique to perform a targeted attack and can we use this knowledge to estimate the likelihood of the attacker's success?

2. Can we predict the vulnerability of the anomaly detection technique to the attack. Can we quantify the degradation in the performance of the system?

We hope that conducting a formal analysis of a poisoning attack on an anomaly detection technique will help us to better understand vulnerabilities in anomaly detection methods, resulting in more effective and robust anomaly detection approaches. By quantifying the degradation in the performance of the system we hope to help the defender to better identify when an attack is underway.

The rest of this paper is organized as follows: Section 2 discusses related work under two spectrums, Anomaly Detection and Adversarial Machine Learning. Section 3 develops a framework for the adversary to manipulate a naive learning algorithm and the impact on DBSCAN. Section 4 describes the experiments for validating the model and discusses the results. Finally, Section 5 presents conclusions and future work.

2. RELATED WORK

2.1 Anomaly Detection

In this section we briefly review clustering based anomaly detection techniques. ML based anomaly detection techniques consist of two phases, training and testing [26]. In the training phase, models of normal behavior are derived from unlabelled or labelled training data. In the testing phase, the models learned are queried to identify whether the new data is anomalous or not. There are many approaches to anomaly detection including classification, clustering, nearest neighbor, statistical, information theoretic and spectral [11]. Clustering based anomaly detection is primarily an unsupervised technique which groups similar patterns together.

DBSCAN [14] is a density based clustering algorithm. The general idea behind this technique is to continue growing the given cluster as long as the density in the neighborhood exceeds some threshold. The points that do not belong to any cluster are treated as anomalies and are further investigated.

2.2 Adversarial Machine Learning

We now briefly review, without a claim for completeness, some previous approaches to security analysis of machine learning algorithms. Adversarial machine learning is the branch intersecting between machine learning and security that aims to study the effectiveness of a machine learning algorithm against an adversary and to learn the capabilities and limitations of the attacker. Huang et al., [18] gave a taxonomy of attacks against a machine learning system. They categorized the attacks based on influence or the capability of an attacker (causative and exploratory), security violation (integrity, availability and privacy) and specificity defined as attackers intention (targeted and indiscriminate). They have presented a boiling frog attack for anomalous traffic detection in which they inject the training data with chaff every week so that the detector gets acclimated to chaff which eventually results in compromising the integrity of the system by increasing the false negatives.

Kloft et al., [20] have analyzed the online centroid based anomaly detection technique in the presence of an adversary. They have formalized the learning and the attack process, derived an optimal attack policy and theoretical bounds on the efficacy of a poisoning attack in perfect knowledge and limited knowledge scenarios. Biggio et al., [6] have used single linkage hierarchical clustering to demonstrate obfuscation and poisoning attacks, evaluating the security of clustering algorithms in adversarial settings. Nelson et al., [23] have developed a model to analyze the efficacy of poisoning attacks and demonstrate the feasibility of the attacks. Fogla et al., [15] have created polymorphic instances of network packets. This ensures that the statistics of attack packet(s) matches the normal traffic profile, thereby subverting the anomaly detection approach. Rubinstein et al., [27] have applied the boiling frog strategy to poison a PCA subspace anomaly detector and have shown that only a moderate amount of poisoned data can substantially decrease the efficacy of the detector.

Newsome et al., [24] have developed a red herring attack that involves poisoning the dataset with spurious features and can mislead the signature generation for malware detection. Dalvi et al., [13] have analyzed bayesian classification for robustness against adversarial impact. The adversarial classification is considered as a game between an attacker and a learner and they have developed a classifier which is optimal given the adversary's optimal attack strategy.

3. FRAMEWORK

The main goal of this work is to determine how vulnerable an anomaly detection technique is likely to be when subjected to an adversarial attack. We want to provide a quantitative framework to do a "what-if" security analysis of the anomaly detection technique. We do this by modeling a near-optimal adversary strategy (in fact optimal when the adversary knows only data distributions and not exact data points), and show how this changes the expectation that an anomalous "adversary point" will be detected. We now discuss the attack model more formally, and look specifically at modeling the impact on DBScan.

3.1 Attack Model

To analyze the vulnerability or the security of a system we need to identify the adversary's goals and its capabilities. To this end, we rely on the taxonomy specified in [3]. We present an attack model that examines a causative attack with the end goal of obfuscating the adversary's data point. The model yields a near-optimal attack strategy and demonstrates the impact on the anomaly detection technique.

3.1.1 Adversary's goal

A clustering algorithm can be formalized as a function f mapping a data set $\mathcal{N}$ to a clustering result C. We define the adversary's goals in terms of attack specificity and security violation as in [3]. We assume that the attack specificity is indiscriminate as the attack targets the entire data set Security violations can affect the integrity or availability of a system. Integrity violations comprise of malicious points

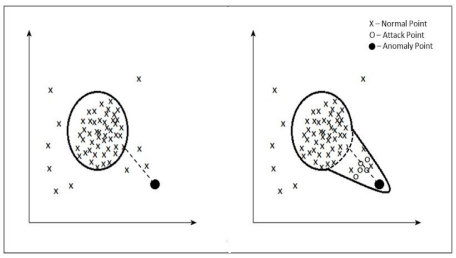

Figure 1: Expansion of the cluster as attack points are added

being classified as normal (false negatives). Availability violation results in misclassifications (false positives and false negatives) so as to render the system unusable. We assume that the attacker wants to perform an integrity violation by including the anomaly or the attack point in the target cluster C. Figure 1 depicts the expansion of the cluster as a consequence of an attack on the anomaly detection technique.

3.1.2 Adversary's knowledge

This is defined as the extent to which an adversary knows about the anomaly detection technique being applied to the system. This encompasses the anomaly detection algorithm, the training dataset and the parameters of the model. The basic assumption while designing a secure system should be that "the enemy knows the system"[2]. Hence, we assume that the adversary has knowledge of the training algorithm and in many cases partial or complete information about the training set, such as its distribution. For example, the attacker may have the ability to eavesdrop on all network traffic over the period of time in which the learner gathers training data. A perfect knowledge scenario exists when the attacker has complete information of the training algorithm and its parameters, dataset and features. In a limited knowledge scenario, we assume that the adversary has partial information about the dataset(e.g.,distributions) but not the dataset. We further assume that the adversary knows the training algorithm and its parameters.

3.1.3 Adversary's Capability

Adversary's capability defines the extent to which the adversary can control the clustering process by manipulating the data set. We assume that the adversary can poison the dataset so as to launch an obfuscation attack. This is realistic in several practical cases, e.g., in the case of spam filtering[18], where the adversary may easily send (a few) samples without having access to the rest of the data. In general we assume that the attacker can generate arbitrary data points for poisoning the training set. We also bound the worst case scenario of the adversary's effort in terms of the maximum number of points that are needed for the attack to succeed in Section 3.2.

[2]This is in accordance with the current security practices of assuming a strong adversary when building a defense system. The underlying tenant of assuming a strong adversary follows the Kerchoff's principle.

3.1.4 Attack Strategy

Once the adversary's goals and capabilities are defined, we can develop an optimal strategy that specifies how to manipulate the dataset so as to meet the adversary's goal of obfuscating the anomaly. This can be achieved by making the neighborhood of the anomaly point denser so that it "looks-like" a normal data point. This is justified under the assumption that a normal datapoint belongs to a larger and a denser cluster. Formally, the strategy can be defined as minimizing the number of attack points that have to be added so as to include the anomaly in a given existing cluster. In formal terms, let $\mathcal{A}$ be the set of attack points, a be the anomaly point, and C be the target cluster. The adversary strategy can be formulated as

$$min|A|, \text{ s.t. } a \in C$$

3.2 Case Study - DBSCAN

In this section we model the impact of the adversary's attack on DBSCAN.[3] To do so, we first derive a near-optimal attack strategy given the underlying properties of the DBSCAN anomaly detection technique. We then model the vulnerability of the technique by studying the change in the evasion success of an adversary as an adversary's control over the training set size increase. The goal of the adversary is to evade detection by expanding the cluster so that the adversary's point (anomaly) is included in the cluster. The notations used in this paper are summarized in Table 1.

3.2.1 Preliminaries

For completeness, we give a few definitions of the DBSCAN algorithm. DBSCAN groups together points that are closely packed together marking low density region points as outliers. For the purpose of DBSCAN the points are classified as core object, (density-)reachable objects, and borders, as follows:

DEFINITION 1. *Core Object - A core object o w.r.t ϵ and α is a data point such that $|N_\epsilon(o)| \geq \alpha$, where $N_\epsilon(o)$ is the ϵ neighborhood of the object o.*

DEFINITION 2. *Directly density reachable - An object p is directly density reachable from o if o is a core object and $p \in N_\epsilon(o)$*

DEFINITION 3. *Density reachable - An object p is density reachable from o if there exists a path $p_1, p_2, ..., p_n$, where $p_1 = p$ and $p_n = o$ such that p_{i+1} is directly density reachable from p_i and $p_2 ... p_n$ are core objects.*

DEFINITION 4. *Border Object - An object p is a border object if it is not a core object but it is density reachable from another core object.*

DEFINITION 5. *Density Based cluster - A cluster C is a non empty subset of $\mathcal{D}$ such that it satisfies the following properties -*

1. *$\forall$ (p,q) if $q \in C$ and p is density reachable from C then $p \in C$.*
2. *$\forall (p, q) \in C$, p and q are density connected.*

[3]We have used DBSCAN as our algorithm for case study as it can identify clusters of varying shapes and sizes and works well with low dimensional data.

Table 1: Summary of the Notation used in this paper

Notation	Definition
n	dimensionality of the data
$\mathcal{D}$	Set of data points or training-data
$\mathcal{A}$	Set of attack points
a	Anomaly point
$dist_n^k(x,y)$	Distance between $(x^1,...,x^n)$ and $(y^1,...,y^n)$ using L_k metric $= \sum_{i=1}^{n}[(x^i - y^i)^k]^{1/k}$. We have assumed k to be 2, i.e.,, we are assuming the distances to be Euclidean.
$N_\epsilon(o)$	the ϵ neighborhood of the object o
α	Minimum cluster size
$\mathcal{C}$	Target Cluster
Y_1	Gaussian distribution with mean μ and variance σ. It is the set of distances of data points from the cluster centroid
d_{min}	The minimum of all the distances between attack point a and y $s.t. y \in C$
$\mathcal{R}$	A low density region between the anomaly point a and the Target Cluster $\mathcal{C}$
$\mathcal{S}$	A high density region between the centroid of the cluster and the border of the cluster
erf	Error Function

3.2.2 Adversary's optimal attack strategy in Limited Knowledge Scenario

We now present the near- optimal attack strategy (or optimal if the adversary knows only the data distributions). To do so, we exploit the property inherent in DBSCAN that if a point is found to be in a dense part of a cluster, its ϵ-neighborhood is also part of that cluster. Let us assume that $\mathcal{C}$ containing the set of points {Y} is the target cluster that the adversary wants to be included in and let "a" be the anomaly point. We now present the algorithm for the obfuscation attack.

ALGORITHM 1: Adversary's attack strategy in Limited Knowledge Scenario

1. Find $y \in Y$ s.t. $dist_n^k(a,y) = \min(dist_n^k(a,Y))$.
2. Let $d_{min} = \min(dist_n^k(a,y))$.
3. Let i = $\lceil d_{min}/\epsilon \rceil$.
4. In a straight line between a and y place core points at $a_1^1 = \epsilon$, $a_1^2 = 2 * epsilon$ ..... $a_1^i = i * \epsilon$.

We now give a proof of correctness for the above algorithm and then quantify the bound on the adversary's effort required to include the attack point in the cluster. Note, that the attack can be reduced from a multi-dimensional space to a single dimensional space because we are using the L_K metric to measure the distances.

Proof of Correctness: DBSCAN in the ExpandCluster function [14], for a given set of points Y in cluster C includes all the points which are at a distance ϵ from any core point y $\in$ Y. At the first iteration, a_1^1 will be included in C as it is at ϵ distance from y. In the second iteration, a_1^2 will be included in C because it is at ϵ distance from core point a_1^1 and so forth. At i+1 iterations, y will be included in C because of a_1^i.

LEMMA 1. *To add i core points in a straight line, the minimum number of points that need to be added i.e., E[|$\mathcal{A}$|]= $\alpha * \lceil i/2 \rceil$, where A is the set of attack points.*

PROOF. If i = 1 , then the number of points needed to make a core object by definition 1 is α. If i is even, then every consecutive core point's e-neighborhood will intersect, hence $E[|\mathcal{A}|]$ = (i/2)*α. If i is odd, then the $E[|\mathcal{A}|]$ = (i/2 + 1)*α. Hence, $E[|\mathcal{A}|= \alpha * \lceil i/2 \rceil$ □

THEOREM 1. *For any $|\mathcal{D}|$ and n, let $d_{min} = \lceil min(dist_n^k(a,Y)) \rceil$, where*

- *Y is the set of points in the cluster C*
- *a is the anomaly point*
- *$min(dist_n^k(a,Y)$ is the minimum distance from a to the cluster C.*

Then, $|\mathcal{A}| = (\alpha) * \lceil d_{min}/(2 * \epsilon) \rceil$

PROOF. $\forall \boldsymbol{n}$:Dist is a function (L_k metric) which takes a point in Y and a and returns a non-negative number as a result, thereby reducing the n-dimensional attack to 1-dimensional for DBSCAN.

From the above argument and Lemma 1, it immediately follows that $\forall$ n, $|\mathcal{A}| = (\alpha) * \lceil d_{min}/(2 * \epsilon) \rceil$.

$\forall |\boldsymbol{\mathcal{D}}|$: We will use induction to prove that irrespective of the size of the set of data points the size of the set of the attack points for DBSCAN is $(\alpha) * \lceil d_{min}/(2 * \epsilon) \rceil$.

For $|\mathcal{D}| = 1$: Let $d_a = dist_n^k(a,y)$. As y is the only point in the cluster C, from the above argument, we have to add only one core point o so as to include y in the 2^{nd} iteration. From Definition 1, a core object o is a data point such that $|N_\epsilon(o)| \geq \alpha$. In order to make o a core object, we have to add $\alpha - 1$ data points in its ϵ neighborhood. It follows that $|\mathcal{A}| = (\alpha) * \lceil d_{min}/(2 * \epsilon) \rceil$, which includes one core object and $\alpha - 1$ directly density reachable objects.

For $|\mathcal{D}|$ = t: Let us assume it to be true for $\mathcal{D}$ = t and let $d_{min}^t = min(dist_n^k(a,Y))$.

For $|\mathcal{D}|$ = t+1: We assume that a point is added to the Cluster C as our target is to include a in C i.e., a $\in$ C. Let a_{t+1} be the point that has been added and let $d_{min}^{t+1} = dist_n^k(a, Y \bigcup a_{t+1})$.

Case 1: If $d_{min}^{t+1} \geq d_{min}^{t}$ then adding a_{t+1} point to $\mathcal{D}$ does not change the attack points or the attack set size as it is not the minimum distance from the cluster C to y.
Case 2: If $d_{min}^{t+1} < d_{min}^{t}$ then the attack set size changes to $|\mathcal{A}| = \alpha * \lceil d_{min}^{t+1}/(2*\epsilon) \rceil$ from $|\mathcal{A}| = \alpha * \lceil d_{min}^{t}/(2*\epsilon) \rceil$. This argument follows immediately from Lemma 1 and how we choose i in Algorithm 1.

Hence, $\forall |\mathcal{D}|$, $|\mathcal{A}| = (\alpha) * \lceil d_{min}/(2*\epsilon) \rceil$. □

3.2.3 Attacker's optimal strategy in Perfect Knowledge Scenario

In the previous subsection, we presented the attackers optimal attack strategy in a limited knowledge scenario and bounded the effort required by the attacker as a function of the size of the attack dataset. In this section, we aim to analyze whether an adversary can do better when the adversary has knowledge of the training data set. The intuition behind reducing the number of attack points needed by an adversary, is that there is a low density region between the border of the cluster C (that an anomaly point wants to be included in) and the anomaly point. A small note - by low density region we refer to a region of radius ϵ that does not have enough data points to form a cluster. A combination of these data points and the attack points added by the adversary can transform a border object to a core object, thereby reducing the effort required by the adversary in terms of the size of the set of the attack points i.e., $|A|$. Let us define this low density region by $\mathcal{R}$ satisfying the following properties -

1. $\mathcal{R} \subset \mathcal{D}$
2. $\mathcal{R} \notin \mathcal{C}$
3. $\forall r \in \mathcal{R}, dist_n^k(r,a) \leq dist_n^k(y,a) \wedge dist_n^k(r,y) \leq dist_n^k(y,a)$, where y is the border point of $\mathcal{C}$

To quantify the reduction in the effort of the adversary, we have to calculate the expected value of the size of this region. To do so, we propose a model and make a few assumptions about the distributions of the data. We now formally state the assumptions and proceed to calculate the size of $\mathcal{R}$.

Assumptions:

1. The Euclidean distances between the data points and the centroid of the cluster is drawn from a Gaussian distribution $Y_1 \sim N(\mu, \sigma)$ with distances within the range $(\mu \pm 6\sigma^2)$.
2. We assume the distance of the anomaly point from the border of the cluster is $d_{min} = min(dist_n^k(a,y))$ as defined in Section 3.2.2. Let d_{min} be represented as $i * \epsilon$, where i is some constant.
3. Since ϵ is a parameter which represents distance we can rewrite ϵ as $c * \sigma$ where c is some constant.
4. $\forall r \in \mathcal{R}, dist_n^k(r,y) \geq 2\sigma$, where y is the centroid of $\mathcal{C}$. It is a reasonable assumption because of the two sigma effect of Gaussian distributions.

Based on these assumptions we calculate the expected size of $\mathcal{R}$ as follows:

$E|\mathcal{R}| = E_{-2\sigma-\epsilon}^{-2\sigma-2*\epsilon}(Y) + E_{-2\sigma-2*\epsilon}^{-2\sigma-3*\epsilon}(Y) + ... + E_{-2\sigma-(i-1)\epsilon}^{-2\sigma-i*\epsilon}(Y)$,
where $i * \epsilon = d$
$= E_{-2\sigma-c\sigma}^{-2\sigma-2c\sigma}(Y) + E_{-2\sigma-2c\sigma}^{-2\sigma-3c\sigma}(Y) + ... + E_{-2\sigma-(i-1)c\sigma}^{-2\sigma-ic\sigma}(Y)$
$= E_{c_1\sigma}^{c_2\sigma}(Y) + E_{c_2\sigma}^{c_3\sigma}(Y) + ... + E_{c_{i-1}\sigma}^{c_i\sigma}(Y)$,
where $c_1 = (-2-c),\ c_2 = (-2-2c), ...,\ c_i = (-2-ic)$
$= |\mathcal{D}| * Pr((c_1 - c_2) \leq Y \leq (c_1 + c_2))$

$$E|\mathcal{R}| = |\mathcal{D}| * \frac{1}{2}[erf(\frac{c_i - \mu}{\sigma\sqrt{(2)}}) - erf(\frac{c_1 - \mu}{\sigma\sqrt{(2)}})], \quad (1)$$

where erf is the error function.

THEOREM 2. *For any $|\mathcal{D}|$ and n, if an adversary has perfect knowledge about the training data, then $|\mathcal{A}| = (\alpha) * \lceil d_{min}/(2*\epsilon) \rceil - E|\mathcal{R}|$*

PROOF. This immediately follows from how we have defined $\mathcal{R}$ and Theorem 1. □

3.2.4 Vulnerability to the attack

So far, we have developed an attack strategy and have bounded the efforts required by the adversary. We now assess the vulnerability of this attack under this attack strategy by analyzing how the false positives and false negatives change with the increase in the number of attack points. A false positive is any error when an anomaly detector (incorrectly) rejects a benign input; we measure the change in this rate with the increase in the number of attack points, whereas false negative is the term used to describe a network intrusion device's inability to detect true malicious events. To do so, we model the distances of the data points and the attack points from the centroid of $\mathcal{C}$ being derived from Gaussian distributions. The goal of this model is to predict the likelihood that an attacker will succeed given the ability to generate some number of fake points. We now state the assumptions formally for our model and then calculate the expected false positives and the false negatives.

Assumptions:

1. The Euclidean distance's between the data points and the centroid of the cluster is drawn from a Gaussian distribution $Y_1 \sim N(\mu, \sigma)$ with distances within the range $(\mu \pm 6\sigma)$, where $\mu = 0$.
2. We assume that there is only one adversary i.e., there is only one anomaly point.
3. A false positive is any point y s.t. $y \in \mathcal{D}$ and $y \notin \mathcal{C}$.
4. To model false negatives, we assume that a false negative is any point a s.t. $a \in \mathcal{A} \cap \mathcal{C}$.

Based on our assumptions, we can calculate the expected false positives as follows:

$$\mathbb{E}(FP) = \sum_{y \in \mathcal{N}} y * p(y_{fp}) \quad (2)$$

$$p(y_{fp}) = Pr(y \in \mathcal{D} | y \notin \mathcal{C}) \quad (3)$$

To simplify the calculation of $p(y_{fp})$, we assume that there exists a point $\gamma = c'\sigma$ s.t. all the data points sampled from the range $(\mu - \gamma, \mu + \gamma)$, are a part of cluster C. Since we are assuming that the distance between the data points and the centroid of the cluster are drawn from a Gaussian distribution, we can safely say that the density decreases as we move away from the mean of the Gaussian distribution. I.e., the centroid of the cluster has a high density region whereas the border of the cluster is in a low density region.

Let us assume that the border of the cluster is at the point γ. The expected density of this region $\mathcal{S}$ is $\mathbb{E}|\mathcal{S}_\gamma| = |\mathcal{D}| * Pr(-\gamma \leq Y \leq \gamma)$, as $\mu = 0$. A false positive will be any point which is not in this region. Therefore,

$$\mathbb{E}(FP) = |\mathcal{D}| - (|\mathcal{D}| * Pr(-\gamma \leq Y \leq \gamma))$$

$$\mathbb{E}(FP) = |\mathcal{D}| - (|\mathcal{D}| * (\frac{1}{2}[erf(\frac{\gamma - \mu)}{\sigma\sqrt{(}2)}) - erf(\frac{-\gamma - \mu)}{\sigma\sqrt{(}2)})]) \quad (4)$$

For completeness, Algorithm 2 describes the process to calculate the value of γ.

ALGORITHM 2: Calculating the value of γ

1. Let $\gamma = \epsilon$. Therefore, the range is $(-\epsilon, \epsilon)$
2. $\mathbb{E}(S_\gamma) = |\mathcal{D}| * Pr(-\gamma \leq Y \leq \gamma)$
3. ctr = 1
4. while $\mathbb{E}(S_\gamma) \geq \frac{\alpha}{\mathcal{D}}$
   $\gamma = (ctr + 1) * \epsilon$
   range r $= (-\gamma, -\gamma + 2 * \epsilon)$
   $\mathbb{E}(S_\gamma) = |\mathcal{D}| * Pr(Y \in r)$
   ctr++

Proof of Correctness : We are assuming that the distances are drawn from a Gaussian distribution with mean $\mu = 0$, therefore the density of any area is maximum around μ and decreases as we move further away from the mean μ. This ensures that the expected value of the density of a cluster will be maximum around the mean of the Gaussian distribution with its density monotonically decreasing with the increase in the distance from the mean. Also, for every point DBSCAN scans a region of radius ϵ which justifies setting the range as $2*\epsilon$. Based on these two facts we can say that the algorithm correctly computes the value of γ.

Note, Table 2 and Table 3 will show how much the gain is in practice for an adversary when he knows the training data set as opposed to when he knows only the distributions.

We now assess the vulnerability of the attack in terms of false negatives. According to our model, the expected false negatives will be $|\mathcal{A}| + 1$ as we are assuming only one adversary and if the adversary adds an attack point, then according to the attack strategy specified in Section 3.2.2 and Section 3.2.3 it will be a part of $\mathcal{C}$.

4. EXPERIMENTS

In the previous sections, we have bounded the efforts of the adversary and assessed the vulnerability for an obfuscation attack using poisoning and have modeled the impact on DBSCAN. We now apply our theoretical results to the real world domain of intrusion detection and validate our theoretical model.

In our experimental scenario, we assume that there is one adversary whose goal is to disguise some anomaly point to get it past the anomaly detection approach (in our case DBSCAN).We assume the parameters are set to achieve a high detection rate and acceptable false alarm rate on a dataset without poisoning. We then test the ability to detect a single adversary point, and how that detection changes as the adversary poisons the data (we repeat this experiment for multiple adversary points, but we assume a single adversary at any given instant). We have implemented our experiments on two anomaly detection datasets - the KDD Cup '99 intrusion detection dataset[12] and the Yahoo! S5 dataset[30]. For the KDD cup dataset we have focused on the denial of service (dos) attacks (back, neptune and teardrop) and the user to root (u2r) attacks (nmap and ipsweep), as these are the attacks where past anomaly detection efforts have shown success. For each attack we have chosen the minimal feature set as given in [25]. The yahoo dataset consists of real and synthetic time-series based on data collected from user activity and server logs, with tagged anomaly points. From the yahoo dataset we have chosen the A4 benchmark dataset which consists of both anomalies and outliers inserted at random positions.

Table 2: Adversary's effort in terms of the $|\mathcal{A}|$ for the KDD Cup '99 Dataset

Attack Type	Adversary's Actual effort	Adversary's predicted effort in Limited Knowledge scenario	Adversary's predicted effort in Perfect Knowledge Scenario
back (a1)	25	30	23
neptune (a2)	46	59	54
teardrop (a3)	33	60	52
nmap (a4)	61	75	61
ipsweep (a5)	46	69	56

4.1 Experiment 1

In this experiment we aim to estimate the effort required by the adversary in terms of the attack size. We will now give a brief description of the experimental setup for the KDD Cup '99 and the Yahoo Webscope dataset.

4.1.1 KDD Cup '99 Dataset[4]

From this dataset, we have randomly sampled 700 normal data points from the training data set i.e., $|\mathcal{D}| = 700$. We also assume that there are 5 different types of adversaries corresponding to the 5 different attack types (back, neptune, teardrop, nmap and ipsweep). For each attack type we average out our results over 10 different instances. For the dos attack we have set $\alpha = 30$ and $\epsilon = 0.4$. These parameters ensures that every instance of every adversary type is detected i.e., True Negative Rate = 0. For the user to root attacks we have set the parameters as $\alpha = 30$ and $\epsilon = 0.12$. Our results are presented in Table 2 and demonstrate the number of attack points required to carry out the obfuscation attack. The results are also compared to the attack size predicted by the model developed in Section 3.2 in a Perfect Knowledge and Limited Knowledge Scenario.

4.1.2 Yahoo S5 - Anomaly Detection Dataset

We have randomly sampled 700 normal data points from the A4 benchmark dataset We assume that there is one type

[4]There have been issues accepting the KDD Cup'99 dataset as a benchmark in the security community because even a trivial detector can achieve perfect accuracy on this dataset. However, we have used this dataset as we needed to validate our model on a real world dataset. We have also randomly sampled a small subset from the KDD Cup ensuring that there were no duplicates.

Table 3: Adversary's effort in terms of the $|\mathcal{A}|$ for the Yahoo Anomaly Detection Dataset

Adversary's Actual effort	19
Adversary's predicted effort in Limited Knowledge scenario	33
Adversary's predicted effort in Perfect Knowledge Scenario	21

of adversary marked as change-points and have averaged out the results over 10 randomly sampled instances of the adversary. For this anomaly detection dataset, we have set $\alpha = 30$ and $\epsilon = 0.4$ ensuring the True Negative Rate is 0. The results are presented in Table 3.
The results presented in Tables 2 and 3 validate that the effort required by the adversary is a worst case bound and he can always do better than the worst case. However, the efforts predicted for the adversary in a perfect knowledge scenario is comparable to the efforts required by the adversary in the real world situation. Thus, the model in the perfect knowledge scenario can be used as a reasonable indicator of the vulnerability of the method to the poisoning/obfuscation attack. Another interesting observation is that the adversary on an average needs to control only 6% of the training dataset to carry out a targeted attack.

4.2 Experiment 2

We now assess the impact of this attack on DBSCAN by analyzing how the evasion rate of an adversary changes with the increase in the attack size.

4.2.1 KDD Cup '99 Dataset

As mentioned previously, we have 5 different types of adversaries (neptune, back, teardrop, nmap and ipsweep) with the same parameters as described above. We assume a constant false positive rate of 0.02 and the evasion rate to be 0 when the initial training data set is trained. Figure 2 & 3 illustrates the evasion rate for the various attack types and compares it to the rates predicted by the model.

4.2.2 Yahoo S5 - Anomaly Detection Dataset

We have one adversary with the same parameters as defined in Experiment 1. The False Positive Rate is set to be 0.01 and Figure 4 illustrates the evasion rate for the adversary at different attack set sizes and compares it to the evasion rate predicted by the model.

An interesting observation from the graphs is that on an average, an adversary only needs to control 5% of the training data set to increase the chances of evasion from 0 to 80%. We also observe here, that the minimum cluster size (α) effects the size of the training set that the adversary needs to control. The reason is intuitive, with a smaller α the effort required to create a core object will be less as compared to a bigger α. We have assumed the minimum cluster size to be 30 or 4% of the training data size. Of note is that there seems to be a fairly clear percentage of the data the adversary needs to be able to poison, at which the attack is likely to succeed; this is fairly well matched by the predicted rate.

4.3 Experiment 3

We have analyzed the adversary's efforts and the probability of the success of the attack in the previous attack. We

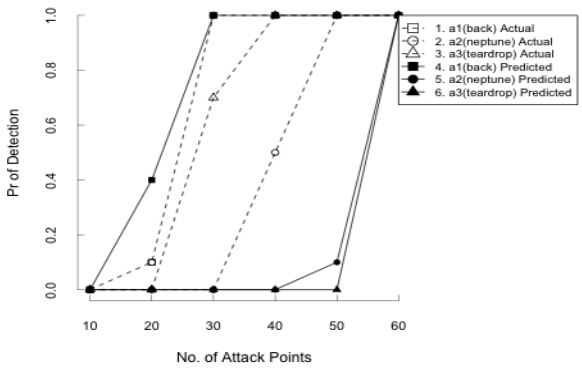

Figure 2: Probability of Evasion vs. Size of the Attack Set for KDD Cup '99 Dataset & DoS attacks

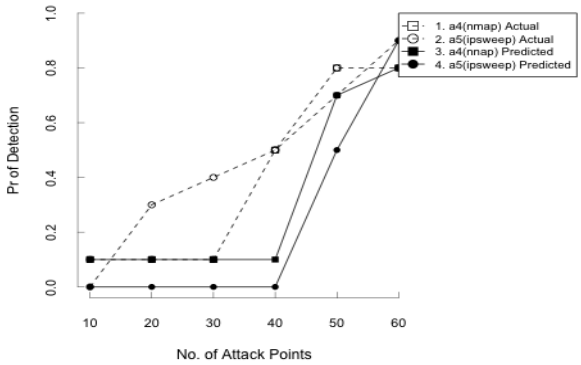

Figure 3: Probability of Evasion vs. Size of the Attack Set for the KDD Cup '99 Dataset & u2r attacks

now assess the vulnerability of this attack by analyzing the change in the false positive rate of the anomaly detection approach and comparing it with the prediction of the model developed in Section 3.2.

4.3.1 KDD Cup '99 Dataset

Our experiments are set up the same as defined in Experiment 1 and Experiment 2. We begin initially with a FPR of 0.024 for the dos attacks (neptune, back and teardrop) and 0.017 for the u2r attacks (nmap and ipsweep) and then illustrate how this changes with the cluster expansion process, as a consequence of adding attack points. Figure 5 displays the results.

4.3.2 Yahoo S5 - Anomaly Detection Dataset

For this dataset, we have a FPR of 0.01 and the changes in the FPR are presented in Table 4.
As it can be seen from Figure 5 and Table 4, the actual and the expected false positive rates decrease with the increase in the attack points. This is counterintuitive because an

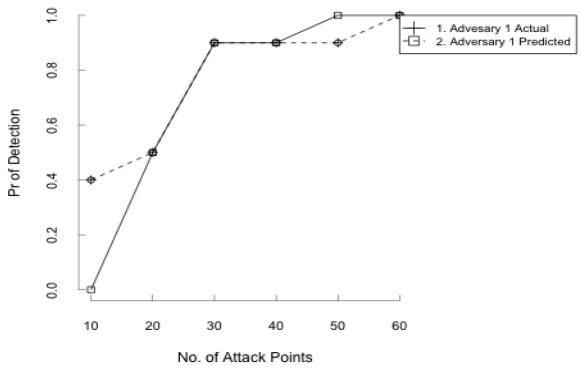

Figure 4: Probability of Evasion vs. Size of the Attack Set for the Yahoo S5 Anomaly Detection Datatset

Table 4: Adversary's vulnerability to Attack for the Yahoo Anomaly Detection Dataset

Adversary's original FPR	0.01
Adversary's FPR after attack	0.007
Adversary's expected FPR	0

attacker would want to increase the false positives so as to render the system unusable. However, our goal here is to perform a targeted attack which results in the expansion of the size of the cluster, ultimately leading to the false positives being included in the cluster resulting in a decrease in the false positive rate. Unfortunately, there is not a huge difference in the actual and the original false positive rates thereby letting the attacker carry out the attack stealthily without an obvious and easily detected increase in the false positive rate.

5. CONCLUSIONS & FUTURE RESEARCH

In this paper, we have addressed the problem of evaluating the security of clustering-based anomaly detection in adversarial conditions by providing an attack strategy and then a framework to bound the adversary's efforts and to assess the vulnerability to an attack. We have provided an optimal strategy in both perfect knowledge and limited knowledge scenarios and have proposed a model to estimate the effort of the adversary and vulnerability to the attack. We have demonstrated with two real world datasets that DBSCAN is vulnerable to an obfuscation attack with a minimal amount of effort from the adversary, and that we can effectively model the vulnerability to such attacks.

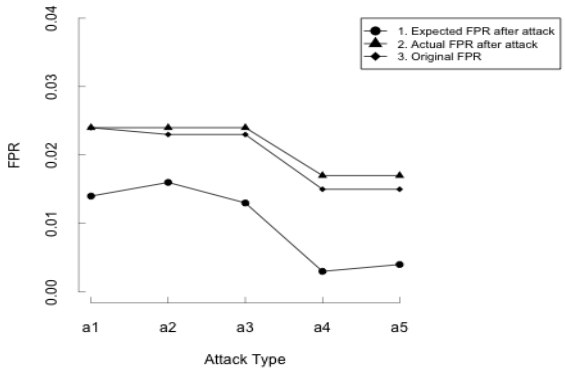

Figure 5: Predicted and Actual FPR for the KDD Cup '99 Dataset

One of the main causes for the vulnerability of DBSCAN is because it relies solely on the distances between two points to determine if they will be included in a cluster or not, thus allowing for an efficient construction of an optimal policy for an obfuscation attack. We can also reasonably assume that density based clustering are more robust to availability attacks than to integrity attacks. Availability attacks render the system unusable by causing a high number of misclassifications (i.e., high false positives and high false negatives). Integrity attacks are the attacks that result in anomaly points being classified as normal points. This is evident as both - the experiments and our model, show a decrease in the false positive rate with an increase in the true negative rate.

One of the main limitations of our model is that we haven't considered a setting where there are multiple adversaries and how they will affect the security of the anomaly detection approach. If multiple adversaries collude, then it could result in a decrease in the effort required by the adversaries. It will be interesting to investigate on how to model collusions between multiple adversaries.

Another limitation of our model is that we have assumed that the adversary knows the anomaly detection technique. It would be interesting to test if uncertainty in the anomaly detection technique makes a poisoning/obfuscation attack significantly more difficult or easier to detect. It would also be interesting to investigate if this attack strategy could be transferred to other anomaly detection methods or whether the same attack samples could be used to alter the learned anomaly detection approach.

In this work, we haven't investigated the detection or the countering of the attack by designing secure algorithms for anomaly detection. Designing an anomaly detection approach which assumes the presence of an adversary can be developed by applying statistically robust techniques or by modeling it as a game between the adversary and the defender. This has been done for supervised prediction techniques like logistic regression where the game was modeled as a static prediction game [8]; it is not clear if the ideas in [8] would be useful for anomaly detection.

6. REFERENCES

[1] A. Alzghoul, M. Löfstrand, and B. Backe. Data stream forecasting for system fault prediction. *Computers & industrial engineering*, 62(4):972–978, 2012.

[2] M. Barreno, B. Nelson, A. D. Joseph, and J. Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

[3] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on*

Information, computer and communications security, pages 16–25. ACM, 2006.

[4] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 387–402. Springer, 2013.

[5] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

[6] B. Biggio, I. Pillai, S. Rota Bulò, D. Ariu, M. Pelillo, and F. Roli. Is data clustering in adversarial settings secure? In *Proceedings of the 2013 ACM workshop on Artificial intelligence and security*, pages 87–98. ACM, 2013.

[7] D. Birant and A. Kut. St-dbscan: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221, 2007.

[8] M. Brückner, C. Kanzow, and T. Scheffer. Static prediction games for adversarial learning problems. *Journal of Machine Learning Research*, 13(Sep):2617–2654, 2012.

[9] M. Çelik, F. Dadaşer-Çelik, and A. Ş. Dokuz. Anomaly detection in temperature data using dbscan algorithm. In *Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on*, pages 91–95. IEEE, 2011.

[10] E. Chan-Tin, D. Feldman, N. Hopper, and Y. Kim. The frog-boiling attack: Limitations of anomaly detection for secure network coordinate systems. In *SecureComm*, pages 448–458. Springer, 2009.

[11] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[12] K. Cup. Intrusion detection data set. *The UCI KDD Archive Information and Computer Science University of California, Irvine. DOI= http://kdd. ics. uci. edu/databases/kddcup99*, 1999.

[13] N. Dalvi, P. Domingos, S. Sanghai, D. Verma, et al. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108. ACM, 2004.

[14] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[15] P. Fogla and W. Lee. Evading network anomaly detection systems: formal reasoning and practical techniques. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 59–68. ACM, 2006.

[16] G. Fortino, R. Giannantonio, R. Gravina, P. Kuryloski, and R. Jafari. Enabling effective programming and flexible management of efficient body sensor network applications. *IEEE Transactions on Human-Machine Systems*, 43(1):115–133, 2013.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[18] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58. ACM, 2011.

[19] M. Kloft and P. Laskov. Online anomaly detection under adversarial impact. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 405–412, 2010.

[20] M. Kloft and P. Laskov. Security analysis of online centroid anomaly detection. *Journal of Machine Learning Research*, 13(Dec):3681–3724, 2012.

[21] P. Laskov and R. Lippmann. Machine learning in adversarial environments. *Machine learning*, 81(2):115–119, 2010.

[22] K. Leung and C. Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, pages 333–342. Australian Computer Society, Inc., 2005.

[23] B. Nelson and A. D. Joseph. Bounding an attack's complexity for a simple learning model. In *Proc. of the First Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML), Saint-Malo, France*, 2006.

[24] J. Newsome, B. Karp, and D. Song. Paragraph: Thwarting signature learning by training maliciously. In *International Workshop on Recent Advances in Intrusion Detection*, pages 81–105. Springer, 2006.

[25] A. A. Olusola, A. S. Oladele, and D. O. Abosede. Analysis of kdd'99 intrusion detection dataset for selection of relevance features. In *Proceedings of the World Congress on Engineering and Computer Science*, volume 1, pages 20–22, 2010.

[26] A. Patcha and J.-M. Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer networks*, 51(12):3448–3470, 2007.

[27] B. I. Rubinstein, B. Nelson, L. Huang, A. D. Joseph, S.-h. Lau, S. Rao, N. Taft, and J. Tygar. Antidote: understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM, 2009.

[28] B. I. Rubinstein, B. Nelson, L. Huang, A. D. Joseph, S.-h. Lau, S. Rao, N. Taft, and J. Tygar. Stealthy poisoning attacks on pca-based anomaly detectors. *ACM SIGMETRICS Performance Evaluation Review*, 37(2):73–74, 2009.

[29] T. M. Thang and J. Kim. The anomaly detection by using dbscan clustering with multiple parameters. In *Information Science and Applications (ICISA), 2011 International Conference on*, pages 1–5. IEEE, 2011.

[30] Yahoo! Webscope. Yahoo! Webscope dataset ydata-labeled-time-series-anomalies-v1_0 [http://labs.yahoo.com/Academic_Relations]. "https://webscope.sandbox.yahoo.com/catalog.php?datatype=s", 2017. Online,Accessed: 2017-12-19.