Faculty of Engineering   
Cairo university   
Computer departmentA logo of a company

Description automatically generated

Statistical and Machine Learning Approaches to IDS

BY CMPlexity  
SUPERVISED BY PROF/MAHA AMIN

|  |  |  |
| --- | --- | --- |
| Names | Code | Roles |
| Amr Hany Sayed | 9230641 | Team Leader |
| Tasneem Ahmed | 9230295 | Team Member |
| Youssef Mohamed | 9231026 | Team Member |
| Hussein Mohamed | 9230345 | Team Member |
| Abdelrahman Medhat | 9230513 | Team Member |
| Kareem Yasser | 9230676 | Team Member |

# Introduction

# Problem Description

In today's digital landscape, cyberattacks have become increasingly sophisticated, rendering traditional intrusion detection systems (IDS) inadequate. These systems often fail to detect novel or subtle threats due to reliance on predefined signatures, resulting in high false positive rates. This project addresses the need for an adaptive, statistical-based IDS capable of detecting anomalies by learning and modeling normal network behavior.

# Background & Problem Statement:

On January 24, 2003, the [W32.SQLExp.Worm](http://securityresponse.symantec.com/avcenter/venc/data/w32.sqlexp.worm.html) (later named Slammer/Sapphire) was released into the wild. This worm exploited a stack-based buffer overflow vulnerability in Microsoft's SQL Server 2000 software (including MSDE 2000). While vulnerabilities affecting Microsoft products are nothing new, the speed at which this worm propagated was extremely novel - scary in fact. The worm was released and within ten minutes it compromised 90% of all vulnerable systems worldwide. Before this incident, worms of this type were merely theoretical, given serious consideration primarily in academia. It takes even the fastest vendors hours or days to produce a signature for rule-based intrusion detection (RBID) systems. In the case of this worm however, a vulnerable network would be compromised in a matter of seconds, much too quickly for even the most diligently updated RBID system. So, what is the solution to a worm that doubles its infection rate every 8.5 seconds? This question is what our research aims to answer, resulting in a statistical-based (also referred to as behavior-based) anomaly detection.

*Intrusion Detection the Statistical Way*

Statistical-based systems (SBIDs) take a different approach to signature-based IDS. The concept of the SBID system is simple: it determines "normal" network activity and then all traffic that falls outside the scope of normal is flagged as anomalous (not normal). SBID systems attempt to learn network traffic patterns on a particular network. This process of traffic analysis continues as long as the SBID system is active, so, assuming network traffic patterns remain constant, the longer the system is on the network, the more accurate it becomes. By analyzing network traffic and processing the information with complex statistical algorithms, SBID systems look for anomalies in the established normal network traffic patterns. All packets are given an anomaly score (indicating the degree of irregularity for the specific event) and if the anomaly score is higher than a certain threshold, the IDS will generate an alert. The key to any SBID system is its ability to learn and distinguish normal from anomalous network activity. The model will flag low-probability events as potential intrusions by monitoring deviations in metrics such as mean, median, and standard deviation.

# Objectives

1. Develop an anomaly detection-based IDS using statistical methods.
2. Identify the effectiveness of various statistical metrics in reducing false positives and negatives.
3. Employ machine learning models to enhance IDS performance in real-world scenarios.

# Statistical Questions:

1. **How can descriptive statistics be used to establish baseline network traffic patterns for an IDS?**
   * Measures like mean, and variance are used to establish "normal" activity.
   * Mean: Average packet size or user login frequency (e.g., if mean packet size = 1,500 bytes).
   * Median: Ensures central tendency isn't skewed by rare, large packets.
   * Variance: Monitors the variability in packet sizes to understand normal fluctuations.
   * **Independent Variable**: Network traffic time intervals
   * **Dependent Variable**: Average packet size
2. **What statistical thresholds are most effective for anomaly detection in network traffic?**

Approach: Use thresholds like:

* + Standard Deviations: Identify packets exceeding, e.g., 3 standard deviations from the mean.
  + Percentiles: Detect anomalies in the top 1% of traffic sizes or login frequencies.
  + Example: If the mean packet size is 1,500 bytes with a standard deviation of 300 bytes, packets over 2,400 bytes (mean + 3×SD) might be flagged.
  + **Independent Variable**: Variance of packet size over time
  + **Dependent Variable**: Detected Anomalies

1. **How does Z-score analysis compare to percentile-based thresholds in detecting network intrusions?**
   * Z-Score: Highlights outliers based on deviation from the mean in terms of standard deviations.
   * Percentile Thresholds: Easier to interpret in real-world contexts. Percentile thresholds provide a simplified way to identify anomalies by focusing on relative rankings within the dataset rather than absolute deviations from the mean (as in Z-scores).
   * Example: A Z-score of 3 might correspond to the 99.87th percentile, but percentiles simplify application (e.g., "flag the top 1%").
   * **Independent Variable**: Time-based packet rate
   * **Dependent Variable**: Anomalous spikes detected
2. **Can moving average and exponential smoothing techniques effectively identify time-based anomalies in network traffic?**
   * Moving Averages: Smooth trends to detect sustained spikes in traffic.
   * Exponential Smoothing: Weights recent data more heavily for real-time trends.
   * Example: Sudden spikes in login attempts might exceed thresholds of the smoothed average, signaling a brute-force attack.
   * Usage:

If the current packet rate exceeds the moving average by a threshold (e.g., 20%), the IDS flags it as suspicious.

* + Advantage:

Detects sustained increases in traffic, such as those caused by Distributed Denial of Service (DDoS) attacks.

* + **Independent Variable**: Protocol type
  + **Dependent Variable**: Frequency of anomalies

**Comparison**

| **Feature** | **Moving Averages** | **Exponential Smoothing** |
| --- | --- | --- |
| **Sensitivity** | Less sensitive to sudden changes. | More responsive to recent changes. |
| **Use Case** | Detect sustained patterns (e.g., DDoS). | Detect sudden anomalies (e.g., spikes). |
| **Ease of Implementation** | Simple to calculate. | Slightly more complex but adaptable. |

**Application in IDS**

* **Moving Averages**:
  + Use for detecting trends in network traffic, e.g., increasing packet size over time.
* **Exponential Smoothing**:
  + Apply for real-time anomaly detection, e.g., bursts of login attempts or packet frequency.

**Practical Example**

* **Network Traffic Data**:

Traffic Volume (MB/sec): [100, 102, 105, 110, 500, 120].

**Moving Average (Window = 3)**:

[100 + 102 + 105]/3 = 102.33 (average smooths gradual changes).

**Exponential Smoothing (α = 0.5)**:

* A sudden jump to 500 is detected quickly due to higher weight on recent values.

In summary:

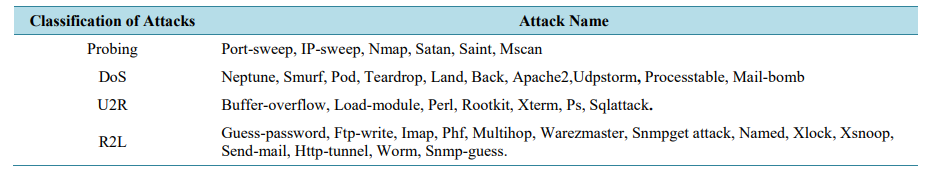
* **Moving averages** are ideal for detecting gradual shifts in patterns.
* **Exponential smoothing** is suited for quick detection of spikes or drops. Both methods are valuable tools in the arsenal of an IDS for identifying anomalies in real-time network traffic.

1. **How can statistical hypothesis testing, such as the Chi-square test, be applied to detect deviations in network traffic patterns?**
   * **Chi-Square Test**: Compares observed vs. expected distributions of packet sizes.
   * **Example**: Test whether an unusual spike in packets > 2,000 bytes is statistically significant.
2. **What is the effectiveness of the t-Test and ANOVA in comparing network traffic features under normal vs. attack conditions?**
   * **t-Test**: Evaluates differences in means between normal and anomalous traffic (e.g., average packet size during attacks vs. normal).
   * **ANOVA**: Analyzes variances across multiple features (e.g., protocols or time periods).
   * **Example**: Compare mean login frequencies across days with attacks vs. normal days.

# Dataset Description and Feature Selection[[1]](#footnote-1)

Key Data Sets

**Dataset 1: RRE-KDD Dataset**

* **Description**: Includes normal traffic and 37 types of attacks (e.g., DoS, U2R).
* **Variables**: Protocol type, service, source bytes, destination bytes, connection flag, anomaly labels.
* **Sample Size**: Training set of 50,000 events; testing set of 20,000 event

**Dataset 2: Unified Host and Network Data Set**

* **Description**: Integrates host-based and network-based features to provide a complete view of system activities. Includes normal behavior and various attack types (e.g., DoS, R2L, U2R, Probing), enabling the detection of both isolated and coordinated attacks.
* **Variables**: User login attempts, file access activities, process execution, protocol types, services, packet sizes, connection durations, and anomaly labels.
* **Sample Size**: Contains over 100,000 events, including 80,000 normal activities and 20,000 labeled malicious events for training and testing purposes.

Intrusion Detection Systems (IDS) must process vast amounts of data to identify potential threats, but the presence of irrelevant and redundant features increases computational complexity and lowers detection rates. Feature selection, also referred to as variable selection, feature reduction, or attribute selection, is a widely used dimensionality reduction technique that reduces the data handled by IDS directly. This reduction is crucial for real-time detection systems where processing speed is critical.

Feature selection methods can be classified into two main approaches: filter and wrapper. Filter models operate independently of any learning algorithm, whereas wrapper models rely on specific learning algorithms, achieving higher classification performance at the expense of increased computational cost. The Random Forest (RF) algorithm effectively integrates feature selection during the classification process by leveraging variable importance measures such as the Gini importance index and permutation importance index.

**Data Set requirement and size**

The sample size is determined by the volume of network traffic and events required to train and evaluate the statistical models. Considering the nature of the research:

1. **Training Data Requirements:**
   * **Purpose:** To train the IDS to identify patterns of "normal" network activity and develop its statistical model.
   * The model will require data to establish a "normal" network activity baseline. This includes:
     + At least **10,000 to 50,000 network events** (packets or user activities) to capture variability in normal traffic.
     + An equal representation of network traffic during peak and off-peak times to ensure robustness.
2. **Testing Data Requirements:**
   * **Purpose:** To evaluate the final performance of the IDS on unseen data.
   * A separate dataset of **5,000 to 20,000 network events** containing both normal and anomalous activities is recommended.
   * The testing data should ideally include a controlled injection of known anomalies to validate the model's ability to detect them.

Total Sample Size:

* + A total sample size of 15,000 to 70,000 events is expected, depending on the complexity of the network and the diversity of traffic patterns.

Additional Considerations:

* + To ensure statistical significance, the dataset should have enough anomalies (e.g., 1-5% of the total events). For example, in a dataset of 50,000 events, 500–2,500 anomalies should be present.

**Evaluation Metrics**

1. **False Positive Rate (FPR):**
   * **What it measures:** The percentage of normal network traffic mistakenly flagged as anomalous.
   * **Why it's important:** A high FPR creates noise and diverts attention from genuine threats, reducing system efficiency. Lowering FPR is crucial for ensuring trust in the IDS.
2. **False Negative Rate (FNR):**
   * **What it measures:** The percentage of actual anomalies that the system fails to detect.
   * **Why it's important:** A high FNR indicates the IDS is missing threats, making it unreliable in detecting critical attacks. Minimizing FNR is vital for securing the network.
3. **Accuracy:**
   * **What it measures:** The overall percentage of correctly classified events (normal or anomalous).
   * **Why it's important:** While it gives a general performance measure, it can be misleading if the dataset is imbalanced (e.g., far more normal traffic than anomalies).
4. **Precision:**
   * **What it measures:** The proportion of detected anomalies that are actually true anomalies.
   * **Why it's important:** High precision ensures the alerts generated by the system are meaningful, minimizing false alarms.

**List of total features and their descriptions in the RREKDD**

Note that: C stands for continuous and D for discrete

****

**Our study focuses on the following features**

|  |  |  |  |
| --- | --- | --- | --- |
| S. No | Feature | Description | Data Type |
| 1 | Duration | Duration of the connection. | C |
| 2 | Protocol type | Connection protocol | D |
| 3 | Service | Destination service | D |
| 4 | Flag | Status flag of the connection | D |
| 5 | Source bytes | Bytes sent from source to destination | C |
| 6 | Destination bytes | Bytes sent from destination to source | C |
| 8 | Wrong fragment | Number of wrong fragments | C |
| 10 | Hot | Number of “hot” indicators | C |
| 11 | Failed Login | Logins number of failed logins | C |
| 12 | Logged in | 1 if successfully logged in; 0 otherwise | D |
| 13 | Compromised | Number of “compromised” conditions | C |
| 14 | Root shell | 1 if root shell is obtained; 0 otherwise | C |
| 22 | Is guest login | 1 if the login is a “guest” login; 0 otherwise | D |
| 23 | Count | Number of connections to the same host as the current connection in the past two seconds | C |
| 24 | Srv count | Number of connections to the same service as the current connection in the past two seconds | C |
| 27 | Rerror rate | % of connections that have “REJ” errors | C |
| 29 | Same srv rate | % of connections to the same service | C |
| 30 | Diff srv rate | % of connections to different services | C |
| 32 | Dst host count | Count of connections having the same destination host | C |
| 34 | Dst host same srv rate | % of connections having the same destination host and using the same service | C |
| 35 | Dst host diff srv rate | % of different services on the current host | C |
| 36 | Dst host same src port rate | % of connections to the current host having the same source port | C |
| 37 | Dst host srv diff host rate | % of connections to the same service coming from different hosts | C |
| 38 | Dst host serror rate | % of connections to the current host that have an S0 error | C |
| 40 | Dst host rerror rate | % of connections to the current host that have an RST error | C |

**Descriptive statistical analysis plots**

**A graph with numbers and points

Description automatically generated with medium confidenceA white rectangular object with black lines

Description automatically generated with medium confidenceA group of rectangular objects with text

Description automatically generated with medium confidenceA close-up of several colored squares

Description automatically generated**

Chi-Square Test

# Overview

The **Chi-Square Test** is a statistical method used to analyze categorical data. It determines:

* Whether there is a significant association between categorical variables (*Chi-Square Test of Independence*).
* Whether an observed distribution matches an expected distribution (*Chi-Square Goodness-of-Fit Test*).

# 1. Chi-Square Test of Independence

## Purpose

This test checks whether two categorical variables are independent.

## Hypotheses

## Steps

1. **Set up a contingency table:** Organize data into rows and columns based on the categories of the variables.
2. **Calculate expected frequencies:**
3. **Compute the Chi-Square statistic:**

* where:
  + : Observed frequency
  + : Expected frequency

1. **Determine the degrees of freedom:**
2. **Compare with the critical value:** Use a Chi-Square table and a chosen significance level ().

## Interpretation

* If or , reject (the variables are dependent).
* Otherwise, fail to reject (the variables are independent).

# 2. Chi-Square Goodness-of-Fit Test

## Purpose

This test determines whether an observed frequency distribution matches an expected distribution.

## Hypotheses

## Steps

1. **Calculate expected frequencies:** Based on theoretical probabilities or a known distribution.
2. **Compute the Chi-Square statistic:**
3. **Determine the degrees of freedom:**
4. **Compare with the critical value:** Use a Chi-Square table and a chosen significance level ().

## Interpretation

* If or , reject (the observed distribution does not fit the expected distribution).
* Otherwise, fail to reject (the observed distribution fits the expected distribution).

# Example: Chi-Square Test of Independence

## Protocal Type Attribute:

### Observed Data (Contingency Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Protocol Type | TCP | UDC | ICMP | Total |
| Normal Anomaly | 40 20 | 10 30 | 35 65 | 85 115 |
| Total | 60 | 40 | 100 | 200 |

**Note:** The data presented in this table is dummy data and does not relate to the actual dataset. It is used purely for illustrative purposes to demonstrate the calculations.

### Calculations

1. **Expected Frequency:**

* Example calculations:
  + For **normal** and **TCP**:
  + For **anomaly** and **UDP**:
  + For **normal** and **ICMP**:

1. **Chi-Square Statistic:**

* Substituting observed () and expected () values:
* Simplify each term:
* Calculate:

1. **Degrees of Freedom:**

### Conclusion

Compare the calculated with the critical value from the Chi-Square table at and :

* If , reject . This indicates an association between protocol type and traffic behavior (*normal* vs. *anomaly*).
* Otherwise, fail to reject , suggesting no association between protocol type and traffic behavior.

## Service Attribute:

Another important attribute in the dataset is **service**, which categorizes network traffic based on the type of service being used. This attribute contains a wide variety of service types, such as:

* http, ftp, telnet, auth, dns, and many others.

The **service** attribute can be analyzed to determine whether there is an association between the traffic type (*normal* vs. *anomaly*) and the services being accessed. This analysis can flag unusual service usage patterns that might indicate network anomalies or intrusions.

### Observed Data (Contingency Table for Service Attribute)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Traffic Type | http | ftp | telnet | auth | dns | other | Total |
| Normal Anomaly | 100 50 | 40 30 | 10 30 | 5 20 | 25 10 | 20 60 | 200 200 |
| Total | 150 | 70 | 40 | 25 | 35 | 80 | 400 |

**Note:** This data is also dummy data and is presented here solely for illustrative purposes.

### Future Calculations

The Chi-Square Test can similarly be applied to this attribute. The steps would involve:

1. Calculating the expected frequencies for each cell in the contingency table using the formula:
2. Computing the Chi-Square statistic:
3. Comparing the calculated value with the critical value to assess independence.

By analyzing the **service** attribute, we can identify potential correlations between certain service types and network anomalies, which might help in detecting specific attack patterns or misuses

## Flag Attribute:

### Observed Data (Contingency Table for Flag Attribute)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Traffic Type | OTH | REJ | RSTO | RSTOS0 | RSTR | S0 | S1 | S2 | S3 | SF | SH | Total |
| Normal Anomaly | 5 10 | 3 15 | 10 20 | 0 1 | 2 3 | 50 40 | 20 10 | 1 5 | 0 1 | 80 70 | 1 0 | 172 175 |
| Total | 15 | 18 | 30 | 1 | 5 | 90 | 30 | 6 | 1 | 150 | 1 | 347 |

**Note:** The data in this table is also dummy data, used for illustrative purposes to explain the potential application of the Chi-Square Test. It does not represent the actual dataset.

### Illustration

This table can be analyzed to determine if there is an association between the traffic type (Normal vs. Anomaly) and the specific flag states observed in the network. The Chi-Square Test would use the observed frequencies in this table to calculate expected frequencies and determine independence between these variables.

**Application of Methods for Statistical Questions**

1. **Central Tendency for Normal Traffic**
   * Apply mean and median to dataset 1. Use deviations to establish thresholds for anomalies.
2. **Chi-Square Test for Protocol-Anomaly Association**
   * Test the independence between protocol type and anomaly labels in dataset 1.
3. **Regression Analysis for Predicting Anomalies**
   * Build a regression model using service and protocol as predictors with dataset 2.
4. **ANOVA for Service-based Anomaly Rates**
   * Compare anomaly detection rates across HTTP, FTP, and Telnet in dataset 1.

Outline for Using Random Forest

# Overview

Random Forest is an ensemble learning technique that combines multiple decision trees to classify network traffic. It is particularly effective for IDS because of its ability to handle large datasets, detect complex patterns, and reduce overfitting through its ensemble approach.

**Purpose**

1. Classify network traffic into normal or anomalous categories.
2. Improve detection accuracy while minimizing false positives and negatives.
3. Leverage the robustness of Random Forest for handling high-dimensional data in real-time.

# Hypotheses

1. **Null Hypothesis (H0)**: Random Forest does not significantly improve the classification of anomalies compared to traditional methods.
2. **Alternative Hypothesis (Ha)**: Random Forest significantly enhances the accuracy and reliability of anomaly detection in IDS.

# Steps

**Step 1: Data Preprocessing**

1. **Input Data**:
   * Dataset: RRE-KDD or real-time network logs.
   * Features: Protocol type, service, source bytes, destination bytes, connection flag, etc.
   * Labels: Normal vs. anomalous traffic.
2. **Cleaning and Encoding**:
   * Handle missing or inconsistent data.
   * Encode categorical variables (e.g., one-hot encoding for "protocol type").
3. **Partitioning**:
   * Split data into training (80%) and testing (20%) sets.

**Step 2: Model Training**

1. **Algorithm**: Train a Random Forest model on the training dataset.
2. **Parameters**:
   * **Number of Trees (no. trees)**: Optimal range is typically 100–500.
   * **Maximum Depth (max depth)**: Limits tree depth to prevent overfitting.
3. **Training Process**:
   * Each tree is trained on a random subset of the data and features.
   * Combine the outputs of individual trees through majority voting for classification.

**Step 3: Model Evaluation**

1. **Metrics for Evaluation**:
   * **Accuracy**: Proportion of correctly classified samples.
   * **Precision**: Percentage of predicted anomalies that are true anomalies.
   * **Recall**: Percentage of true anomalies correctly identified.
   * **F1-Score**: Harmonic mean of precision and recall.
2. **Confusion Matrix**:
   * Evaluate model predictions using True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN).

**Step 4: Anomaly Detection**

1. **Classification**:
   * Label network traffic as "normal" or "anomalous" based on majority votes across trees.
2. **Confidence Scores**:
   * Use class probabilities for threshold-based anomaly detection.
   * Example: If the probability of an anomaly exceeds 0.8, classify it as anomalous.

# Example Calculations

1. **Prediction with Majority Voting**:
   * If 60% of trees classify an input as "anomalous" and 40% as "normal," the final output is "anomalous."
2. **Evaluation Metrics**:
   * **Accuracy**
   * **Precision**
   * **Recall**
   * **F1-Score**

**Comparison with Other Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| Metric | Random Forest | Chi-Square Test | Regression Analysis |
| Classification Power | High | Limited | Moderate |
| Scalability | Excellent | Moderate | Limited |
| Nonlinear Patterns | Captures well | Not applicable | Limited |
| Real-Time Capability | High | Limited | High |

1. Hasan, M.A.M., Nasser, M., Ahmad, S. and Molla, K.I. (2016) Feature Selection for Intrusion Detection

   Using Random Forest. *Journal of Information Security*, **7**, 129-140. http://dx.doi.org/10.4236/jis.2016.73009 [↑](#footnote-ref-1)