# Network Intrusion Detection in an Adversarial Setting

**Problem Statement:** To break classifiers trained for Network Intrusion Detection by supplying them with Adversarial examples.

## Dataset

We use the NSL-KDD dataset. The dataset is based on the original KDD cup 99 dataset, but improved to remove some statistical issues with the dataset. The problems with the KDD cup 99 dataset are as follows -

1. There is a huge number of redundant records for about 78% and 75% are duplicated in the train and test set, respectively.
2. This makes the machine learning quite based.

The NSL-KDD dataset has been used in many Network Intrusion Detection research papers. It provides a good analysis on various machine learning techniques for intrusion detection.
The advantages of using this dataset are -

1. No redundant records in the train set, so the classifier will not produce any biased result
2. No duplicate record in the test set which have better reduction rates.
3. The number of selected records from each difficult level group is inversely proportional to the percentage of records in the original KDD data set.

Although, the data set still suffers from some of the problems in the KDD cup 99 dataset and may not be a perfect representative of existing real networks, but because of the lack of public data sets for network-based IDSs, this dataset can still be applied as an effective benchmark data set to help researchers (and us) compare different intrusion detection methods.

### Dataset Description

Each record in the NSL-KDD dataset has 41 features. The features belong to three major families -

- *Basic features* - The ones that are related to connection information such as hosts, ports, services used and protocols.
- *Traffic features* - The ones that are calculated as an aggregate using a window interval.
- *Content features* - The ones extracted from the packet data or payload and they are related to the content of specific applications or the protocol used.

| No. | Feature | Type | Description |
|---|---|---|---|
| | Basic features of individual TCP connections | | |
| 1 | Duration | Numeric | Duration of the connection |
| 2 | Protocol_type | Nominal | Type of the protocol |
| 3 | Service | Nominal | Network service on the destination |
| 4 | Flag | Nominal | Normal or error status of the connection |
| 5 | Src_bytes | Numeric | # of bytes transferred from source to destination |
| 6 | Dst_bytes | Numeric | # of bytes transferred from destination to source |
| 7 | Land | Binary | 1 if connection is from/to the same host/port; 0 otherwise |
| 8 | Wrong_fragment | Numeric | # of "wrong" fragments |
| 9 | Urgent | Numeric | # of urgent packets (with the urgent bit set) |
| | Content features within a connection suggested by domain knowledge | | |
| 10 | Hot | Numeric | # of "hot" indicators |
| 11 | Num_failed_logins | Numeric | # of failed login attempts |
| 12 | Logged_in | Binary | 1 if successfully logged in; 0 otherwise |
| 13 | Num_compromised | Numeric | # of "compromised" conditions |
| 14 | Root_shell | Binary | 1 if root shell is obtained; 0 otherwise |
| 15 | Su_attempted | Binary | 1 if "su root" command attempted; 0 otherwise |
| 16 | Num_root | Numeric | # of "root" accesses |
| 17 | Num_file_creations | Numeric | # of file creation operations |
| 18 | Num_shells | Numeric | # of shell prompts |
| 19 | Num_access_files | Numeric | # of operations on access control files |
| 20 | Num_outbound_cmds | Numeric | # of outbound commands in an ftp session |
| 21 | Is_hot_login | Binary | 1 if the login belongs to the "hot" list; 0 otherwise |
| 22 | Is_guest_login | Binary | 1 if the login is a "guest" login; 0 otherwise |
| | Traffic features computed using a two-second time window | | |
| 23 | Count | Numeric | # of connections to the same host as the current connection (*Note: The following features refer to these same-host connections.*) |
| 24 | Serror_rate | Numeric | # of connections that have "SYN" errors |
| 25 | Rerror_rate | Numeric | % of connections that have "REJ" errors |
| 26 | Same_srv_rate | Numeric | % of connections to the same service |
| 27 | Diff_srv_rate | Numeric | % of connections to different services |
| 28 | Srv_count | Numeric | % of connections to the same service as the current connection in the past two seconds (*Note: The following features refer to these same-service connections.*) |
| 29 | Srv_serror_rate | Numeric | % of connections that have "SYN" errors |
| 30 | Srv_rerror_rate | Numeric | % of connections that have "REJ" errors |
| 31 | Srv_diff_host_rate | Numeric | % of connections to different hosts |
| | Host based traffic features computed using a two-second time window | | |
| 32 | Dst_host_count | Numeric | # of connections having the same destination host |
| 33 | Dst_host_srv_count | Numeric | # of connections using the same service |
| 34 | Dst_host_same_srv_rate | Numeric | % of connections using the same service |
| 35 | Dst_host_srv_diff_host_rate | Numeric | % of different services on the current host |
| 36 | Dst_host_same_src_port_rate | Numeric | % of connections to the current host having the same src port |
| 37 | Dst_host_srv_diff_host_rate | Numeric | % of connections to the same service coming from different hosts |
| 38 | Dst_host_serror_rate | Numeric | % of connections to the current host that have an S0 error |
| 39 | Dst_host_srv_serror_rate | Numeric | % of connections to the current host that and specified service that have an S0 error |
| 40 | Dst_host_rerror_rate | Numeric | % of connections to the current host that have an RST error |
| 41 | Dst_host_srv_rerror_rate | Numeric | % of connections to the current host and specified service that have an RST error |

**Each record in the NSL-KDD dataset is labeled with either normal or a particular class of attack.** The training data contains 23 traffic classes that include 22 classes of attack and one normal class. The test data contains 38 traffic classes that include 21 attacks classes from the training data, 16 novel attacks, and one normal class.

The attack types are grouped into four categories -

1. DoS (Denial of Service) - Attacks that target availability or prevent legitimate users from accessing information or services.
2. Probe - Attacks that aim at gathering information by scanning or probing the network.
3. U2R (User to Root) - Attacks that attempt to access normal user account and exploit vulnerabilities in the system for privilege escalation.
4. R2L (Remote to Local) - Attacks that attempt to gain unauthorized remote access to a local machine.

## Data Preprocessing

### One-Hot Encoding

The features in the NSL KDD dataset have three data types: nominal, binary and numeric. We use one hot encoding to convert the nominal features, “protocol_type”, “service” and “flag”. Since, “protocol_type” has three types of values - “tcp”, “udp” and “icmp”. So, we convert the column “protocol_type” to “protocol_type_tcp”, “protocol_type_udp” and “protocol_type_icmp”. This helps to convert the nominal values to binary values.

Using one-hot encoding, the feature ‘‘service’’ is transformed to 70 new features, and the feature ‘‘flag’’ to 11 new features. In this way, the 41-feature dataset is mapped to a 122-feature dataset.

| Attack Label | Attack Type |
|---|---|
| Denial of Service (DOS) | Back, Land, Neptune, Pod, Smurf, Teardrop, Apache2, Udpstorm, Processtable, Worm, Mailbomb |
| Probe | Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm |
| Remote to Local (R2L) | Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, ProbHttptunnel, Sendmail, Named |
| User To Root (U2R) | Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps |

Attack Types

### Normalisation

Min-max scaling is used to normalise all the numerical values to values between 0 and 1. This helps to prevent imbalanced results by some classifiers.

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

### Final Summary of Dataset

After one-hot encoding, normalization, and classification of attack types, the problem was transformed to a 5-class classification problem where the 5 labels are ‘‘Normal’’, ‘‘DoS’’, ‘‘Probe’’, ‘‘R2L’’, and ‘‘U2R’’, and the 122 numeric features fall into the range between 0 and 1. The number of samples in the training set is 125,973 and in the test set 22,544.

## Adversarial Attacks

### How Adversarial Attacks Work (YCombinator blog By Emil Mikhailov and Roman Trusov)

Studies by Google Brain show that all Machine Learning classifiers can be tricked to give incorrect predictions. For Adversarial Attacks, we generate input in a specific way to get the wrong results from a model.

Types of Adversarial attacks -

1. **Non-targeted Adversarial attack -** The most general type of attack in which the objective is simply to make the classifier give some output other than the intended output.
2. **Targeted Adversarial attack -** Here the aim is to fool the classifier into predicting a specific class for a given data point (obviously different from the intended class). These are harder to carry out.

**Algorithms**

- **FGSM (Fast Gradient Sign Method) -** The idea is to add some weak noise on every step of optimisation, to drift towards the required class (targeted attack) or away from the actual class (non-targeted attack). This is like an optimisation problem, we optimise the noise to **maximise** the error.
  Good tutorial -
  https://towardsdatascience.com/adversarial-examples-in-deep-learning-be0b08a94953
  We take the derivative of the loss function w.r.t x, since the parameters of the model are already computed and y is also fixed.

  $$\nabla_x L(\theta, x, y)$$

  We take a very small value, epsilon (ε) and multiply it with the gradient. This is the perturbation we introduce.

  $$\eta = \varepsilon \ \mathrm{sign}(\nabla_x L(\theta, x, y))$$

  So this perturbation is added to the original data to generate adversarial examples.

  $$x_{adv} = x + \eta$$

  The family of attack where you are able to use compute gradients using the target model are called **white-box attacks**.