

Protecting Telecommunications Infrastructure against DDoS Attacks through Artificial Intelligence Tools

Conf.dr.ing. Daniel ZINCA
daniel.zinca@com.utcluj.ro

DEPARTMENT

UC Labs
Unified Communications Laboratories

Agenda

1. DoS/DDoS attacks
2. IDS/IPS
3. Datasets
4. GMLMs (Generative Machine Learning Models)
5. GANs (Generative Adversarial Networks)
6. GMLMs applications to Intrusion Detection
7. GAN-Based Data Augmentation for Existing Datasets
8. Performance analysis
9. Conclusions
10. References

1. DOS/DDoS

DoS attacks aim to disrupt the availability of a target system or network by overwhelming it with excessive traffic or resource requests originating from a single source.

Characteristics:

- Single attacking host
- Flooding or exploiting vulnerabilities
- Causes service degradation or downtime

Examples:

Smurf, Neptune, Pod, Teardrop, Land

1. DOS/DDoS

DDoS attacks involve multiple compromised devices (botnets) simultaneously targeting a victim to exhaust resources and disrupt service availability.

Characteristics:

- Multiple attacking hosts (distributed sources)
- Coordinated traffic flooding
- More complex to detect and mitigate
- Can leverage amplification techniques

2. IDS/IPS

It is considered that firewalls can eliminate 80% of the attacks in a network

New vulnerabilities are permanently discovered and new attacks must be mitigated (Zero-day Attacks)

Networks must be able to recognize and mitigate new threats

Solutions:

- IDS Intrusion Detection Systems
- IPS Intrusion Prevention Systems

IDS/IPS use signatures (set of rules) to detect/protect from intrusions

IDS Detection Types

- Pattern-based detection
- Policy-based detection
- Anomaly-based detection

Signature Attributes

- Signatures, representing unique patterns in malicious traffic, allow for the identification of specific malware and anomalous protocol behavior
- Signatures enable IDS/IPS sensors to detect known attacks within network packets and respond according to predefined rules
- Signatures have three distinctive attributes:
  - **Type** - Atomic or Composite
  - **Trigger** - the alarm, packet/s characteristics
  - **Action** - What the IDS/IPS will do

Evaluating alerts

- **True positive** - (Desirable) The IDS generates an alarm when it detected a known attack traffic.
- **True negative** - (Desirable) Normal network traffic does not generate an alarm.
- **False positive** - (Undesirable) An IDS generates an alarm after processing normal user traffic that should not have triggered an alarm. Must be tuned to change these alarm types to true negatives.
- **False negative** - (Dangerous) An IPS fails to generate an alarm when known attacks are not being detected. Malicious traffic was not detected by the IDS in place. Must be tuned to generate true positive alarms.

IDS/IPS Problem Statement

- **Limitations of signature-based IDS/IPS**
- **Challenges:**
- zero-day (unknown) attacks
- Not scalable for Increasing traffic and enhanced list of signatures
- **Questions when considering an AI/ML IDS implementation:**
- Are discriminative models (SVM, RF, CNNs) able to detect anomalies?
- Can a Generative Model enhance the IDS performance?

3.Datasets

1. NLS-KDD
2. CICDDoS2019

3.1.The NSL-KDD Dataset

Source:
Canadian Institute of Cybersecurity https://www.unb.ca/

Attacks:

- DoS Denial Of Service
- Probe
- USR User To Root
- R2L: Remote to Local

DoS attacks in the Dataset:

- Smurf, Neptune, Pod, Teardrop, Land, Back

3.1.The NSL-KDD Dataset

41 features grouped in 4 Feature Categories:

- Basic Features
  - Protocol type/Service/Bytes/…
- Content-related Features
  - Logged-in/Root shell /…
- Time-related Features
  - Servercount/Serror_rate /…
- Host-based traffic Features
  - Dst_host_diff_srv_rate/ Dst_Host_count /…

The NSL-KDD Dataset

Dataset	Number of Records:					
	Total	Normal	DoS	Probe	U2R	R2L
KDDTrain+20%	25192	13449 (53%)	9234 (37%)	2289 (9.16%)	11 (0.04%)	209 (0.8%)
KDDTrain+	125973	67343 (53%)	45927 (37%)	11656 (9.11%)	52 (0.04%)	995 (0.85%)
KDDTest+	22544	9711 (43%)	7458 (33%)	2421 (11%)	200 (0.9%)	2654 (12.1%)

Problem: imbalance for the U2R and R2L records

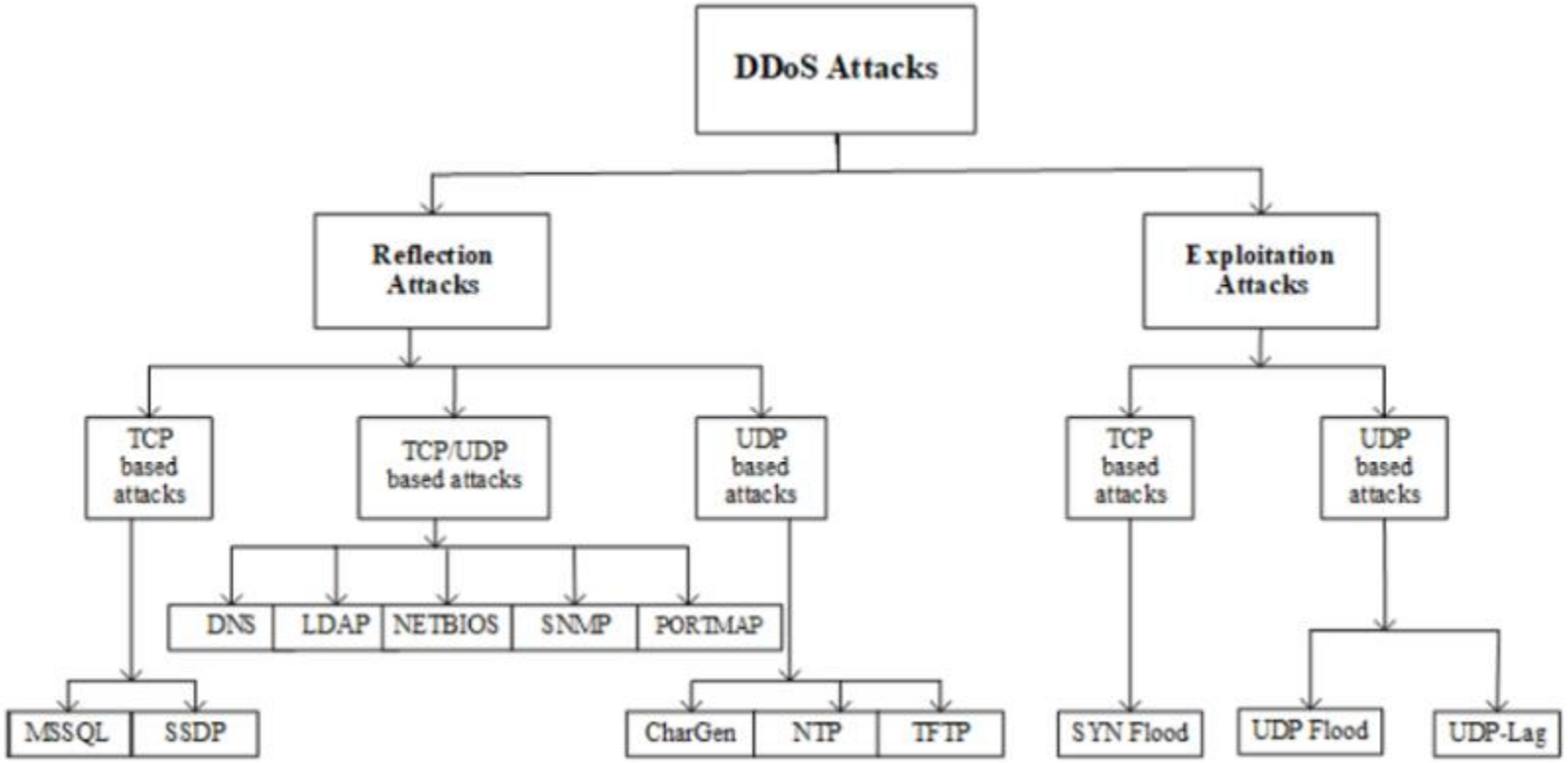
The CICDDoS2019 Dataset

- DDoS (Distributed Denial of Service) attacks on Layers 4 and 7.
- Captured in two different days
- 80 features were extracted (using the CICFlowMeter v3 tool)
- Contains the entire traffic in pcap format + the csv files with the extracted features
- Each csv file was named according to the attack present in the majority of the records, along with some benign traffic
- https://www.unb.ca/cic/datasets/ddos-2019.html

Attacks in the CICDDoS2019 Dataset

Day 1	Day 2
Syn (TCP) **UDP** **UDPLag**	Syn (TCP) DrDoS UDP UDPLag
LDAP **MSSQL** **NetBIOS** **Portmap**	DrDoS LDAP DrDoS MSSQL DrDoS NeBIOS DrDos NTP DrDoS SNMP DrDoS SSDP DrDoS DNS TFTP

Attacks in the CICDDoS2019 Dataset



Source:
https://www.unb.ca/cic/datasets/ddos-2019.html

Flooding Attacks

Attempt to exhaust the resources available at the targeted entity, by sending a large number of packets, flooding the channel or the server resources.

One subcategory is represented by an exploitation attack where a “vulnerability” in the protocol design is used, like in the case of the TCP SYN-Flood.

Reflection and amplification Attacks

The attacker is spoofing packets having as source address the target device
This determines other computers to send back packets to the victim's IP address

This attack category is called DrDoS (Distributed Reflection Denial of Service)

CICDDoS2019 Dataset

Aspects that can lead to a lower performance when using the dataset:

1.In the case of some protocols, the file from one day contains records for a different attack (for example a flooding attack) than the records targeting the same protocol in the other file (where a DrDoS attack is recorded).

2.The behavior of a Layer 4 (L4) attack is different than the behavior of a Layer 7 (L7)-based one.

Significant occurrence of identical packets differentiated only by their timestamp

4.GMLMs
(Generative Machine Learning Models)

A class of machine learning models designed to generate new data samples that resemble a given training dataset by **learning** the underlying data distribution.

- probabilistic/statistical models
- deep learning models

GMLM Variants

A. Probabilistic/Statistical Generative Models

- Naïve Bayes
- GMM (Gaussian Mixture Models)
- HMM (Hidden Markov Models)

B. Deep Learning-Based Generative Models

- GANs (Generative Adversarial Networks)
- VAEs (Variational Autoencoders)
- Autoregressive Models
- Diffusion Models
- Flow-Based Models

Probabilistic/Statistical Generative Models

Naïve Bayes - Probabilistic model assuming feature independence to generate class-conditional data

GMM (Gaussian Mixture Models) - Represent data as a mixture of multiple Gaussian distributions for sampling new points

HMM (Hidden Markov Models) - Model sequential data probabilistically with hidden states

Deep Learning-Based Generative Models

GANs (Generative Adversarial Networks) - Uses adversarial training to generate realistic data samples

VAEs (Variational Autoencoders) - Encode data into latent space and decode probabilistically to generate new samples

Autoregressive Models - Sequentially generate data, modeling conditional distributions (e.g., GPT)

Diffusion Models - Transform noise into structured data via iterative refinement

Flow-Based Models - Learn invertible transformations for exact likelihood estimation and sample generation

Variational Autoencoders (VAEs)

Definition:
Probabilistic generative models that encode input data into a latent space and decode from it to generate new data samples, learning the underlying data distribution

4 Key Components:

- **Encoder**
- **Latent Space**
- **Decoder**
- **Loss Function**

VAE (Variational Autoencoder)

Encoder: compresses data x into a latent vector z $q(z \mid x)$.
Latent Space: compact representation of data with probabilistic sampling
Decoder: reconstructs data $\hat{x}$ from z $p(x \mid z)$.
Loss function: combines Reconstruction Loss with KL (Kullback-Leibler) Divergence Loss
Trained to **maximize the likelihood** of the data while keeping latent space continuous using the *KL divergence* regularization term.
The goal is to learn a smooth latent space where similar points generate similar samples. Learns latent distribution $P(z \mid x)$ and reconstructs data

Application in Intrusion Detection: Anomaly detection: Learn normal traffic patterns → detect anomalies as high reconstruction error

5.GANs

A GAN consists of two neural networks with connected outputs to each other:

Generator tries to generate synthetic (fake) data (from random noise) indistinguishable from real data.

Discriminator tries to distinguish real from generated(fake) data.

These two networks are trained in an adversarial mode:

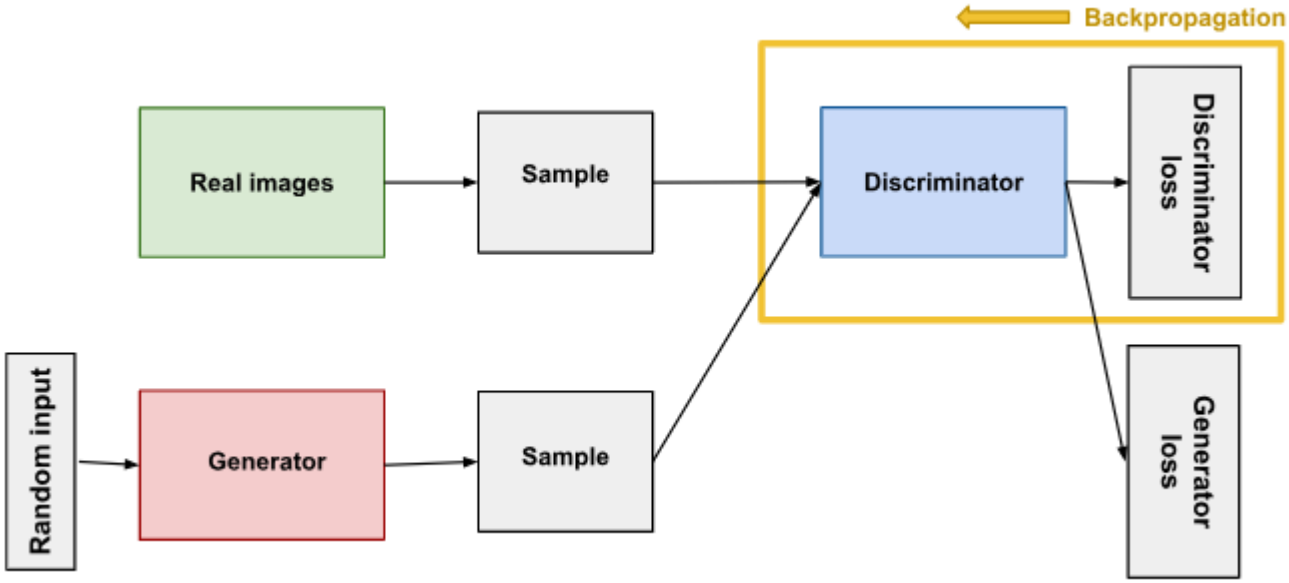
G tries to deceive D

D tries to classify correctly the synthetic data

Initially developed for image generation tasks

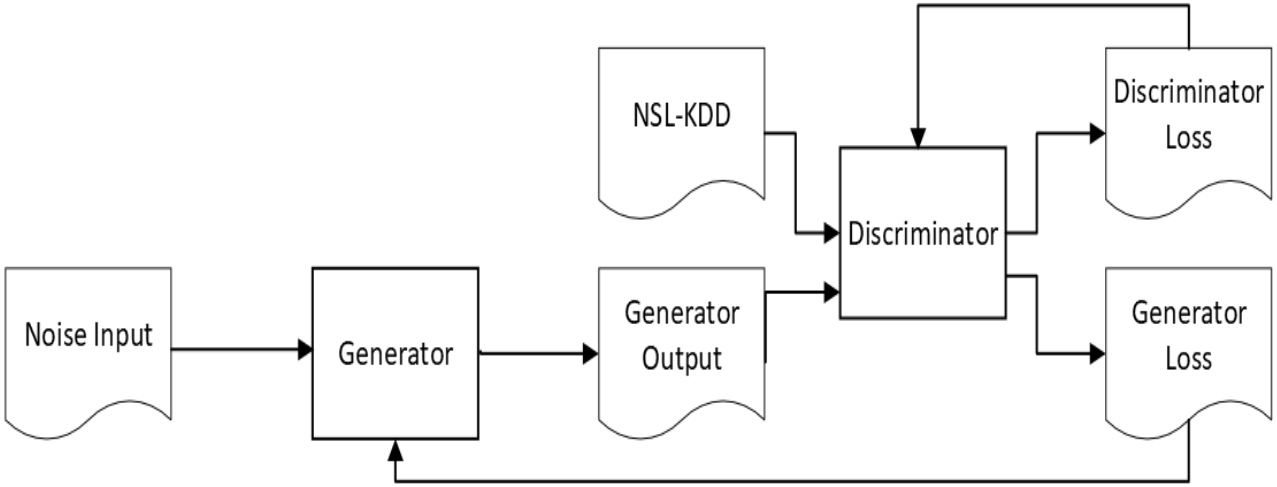
Original Paper by Ian Goodfellow et al (2014)

GANs



Source: Google developer https://developers.google.com/static/machine-learning/gan/images/gan_diagram_discriminator.svg

GANs applied in Intrusion Detection



Source: Mari, Zinca, Dobrota - Development of a Machine-Learning Intrusion Detection System and Testing of Its Performance Using a Generative Adversarial Network

GANs vs VAEs in Intrusion Detection

Both VAE and GAN are generative models, but they differ in how they learn data distributions.

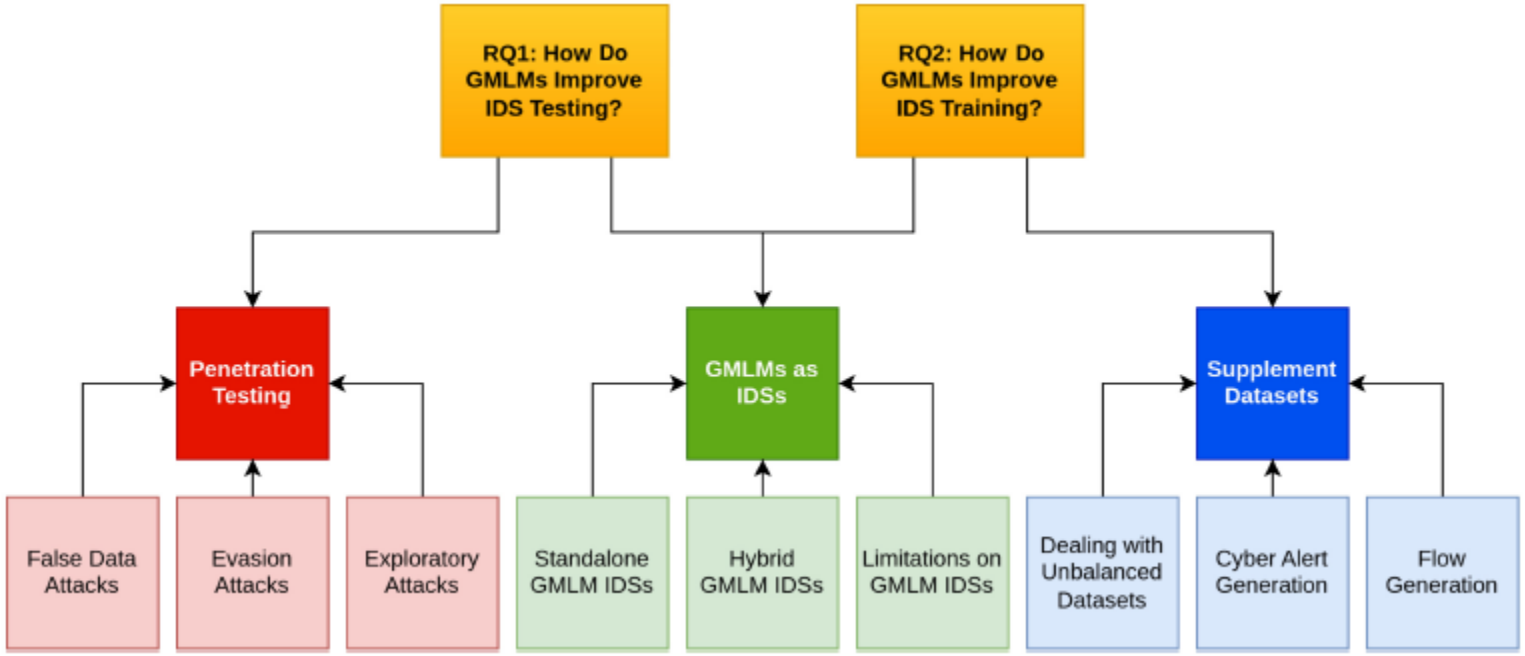
VAEs reconstruct data through probabilistic encoding, while GANs generate data through adversarial learning.

Best applications for Intrusion detection:

VAEs are better for anomaly detection

GANs are better for data augmentation (help balance and expand datasets).

6.GMLM applications to Intrusion Detection



Source: Applying Generative Machine Learning to Intrusion Detection: A Systematic Mapping Study and Review

GMLMs Applications to Intrusion Detection

Dataset improvement : Generate synthetic records for rare attack types to balance dataset
Penetration Testing: Simulate new patterns that may not appear in the training set.

GMLMs as IDS: Balanced training data can improve classifier robustness.

7. GAN-Based Data Augmentation for Existing Datasets

Generate synthetic records for rare attack types to balance dataset

Example: for U2R and R2L in the NSL-KDD dataset

GAN-Based Data Augmentation for Existing Datasets

Table 1. The functional features of each attack category.

Attack	Intrinsic	Content	Time-based traffic	Host-based traffic
Probe	✓		✓	✓
DoS	✓		✓	
U2R	✓	✓		
R2L	✓	✓		

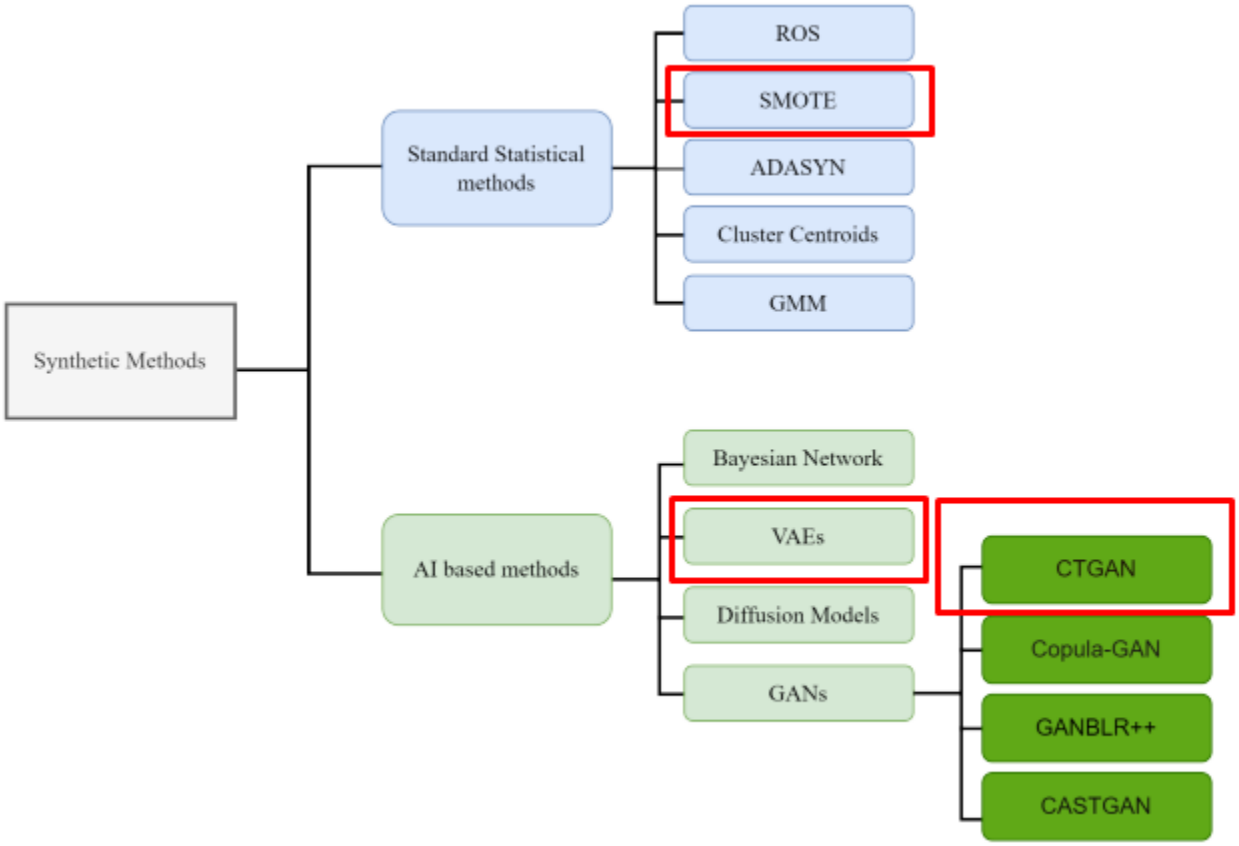
Sample generation must retain the functional features of the attack. The functional features must be kept unchanged to preserve attack functionality (to simulate realistic attack behavior).

Source: IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection

NSL-KDD Feature list

Basic Features (1–9): basic packet information such as ‘duration’, ‘Protocol type’, ‘service’, and ‘flag’.
Content Features (10–22): They are based on packet content information and are used to identify **R2L and U2R attacks**, such as multiple failed login attempts.
Time-based Features (23–31): Including features like ‘count’ and ‘srv count’; they are calculated within a 2 s window. These features represent patterns in network connections.
Host-based Features (32–41): With features such as ‘dst host count’ and ‘dst host srv count’, these are calculated over windows spanning more than 2 s and represent attack patterns from destination to host.

Synthetic Data Generation



Source Synthetic Network Traffic Data Generation: A Comparative Study
Note: in red, the sections implemented in the Lab

Example: NSL-KDD Data Augmentation

Purpose:
Augmentation of the R2L and U2R classes.
In the original training dataset, U2R has 52 records and R2L has 995 records (biased)

Steps:
1.use SMOTE to increase each class to have at least 1000 records.
2.select the columns that are relevant to U2R and R2L attacks (general 1-9 and content 10-22 columns from the NSL-KDD)
3.use CTGAN to generate 1000 records for R2L and 1000 for U2R fot that specific columns. For the rest of the columns, add random values from the real samples of the same attack type
4.Measure the performance (the difference between real and synthetic data) using CDF, the Kolmogorov-Smirnov test and T-SNE
5.Use VAE for generating samples and compare its performance with GAN

SMOTE Synthetic Minority Oversampling Technique

Selects the minority class sample (a U2R or R2L instance)
Finds its nearest neighbors within the same class;
Interpolates new synthetic samples *between* them in feature space.

This generates new, realistic data points that increase the representation of the minority class.
Purpose in this example:
It helps generative models (e.g., GANs like CTGAN or VAE) learn better feature distributions.

Relevant column selection

```
#only the relevant columns for u2r and r2l attacks will be used with CTGAN
 #columns 1-9:basic features(relevant for all attacks); 10-22:content features(for
u2r and r2l attacks)
cols_for_ctgan = ['duration', 'protocol_type', 'service', 'flag', 'src_bytes',
'dst_bytes','land','wrong_fragment','urgent',
                  'hot','num_failed_logins', 'logged_in', 'root_shell', 'su_attempted',
      'num_file_creations', 'num_shells', 'num_access_files',
'is_host_login','is_guest_login'
      ,'attackCateg']
#subset_data = data[cols_for_ctgan]#without SMOTE
subset_data=subset_balanced[cols_for_ctgan]
```

CTGAN for Data Augmentation

Overview

CTGAN (Conditional Tabular GAN) is a **GAN variant specifically designed for tabular (mixed-type) data**, developed by MIT's Data-to-AI Lab (used in the *SDV – Synthetic Data Vault* library).

CTGAN can handle:

Continuous + categorical features (like NSL-KDD's 41 mixed features).

Highly imbalanced distributions.

Skewed numeric variables via mode-specific normalization.

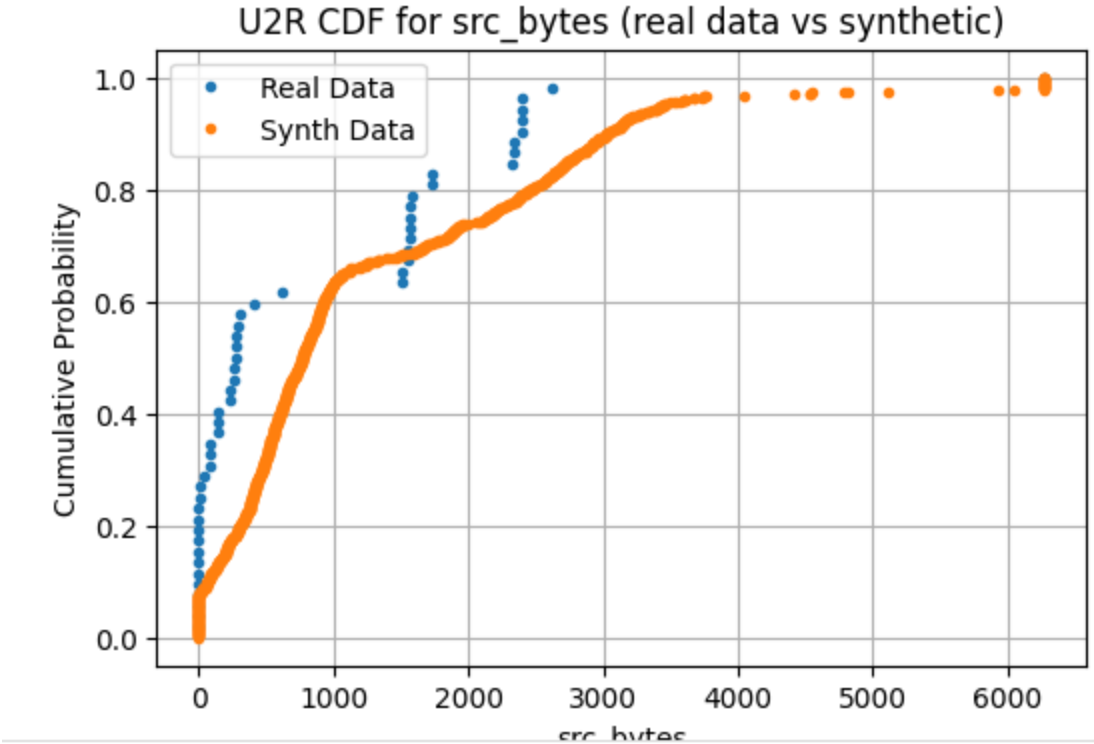
Conditional sampling (e.g., generate only "attack" or "normal" records).

Probabilistic, **can generate** samples with values **outside** the original domain.

Performance Analysis

Visualize how **real and synthetic samples overlap** after GAN training.
Detect **mode collapse** (GAN producing limited variations).
Compare data distribution coverage before and after augmentation.
Explore latent space structure to design conditional GANs.

CTGAN results



comparison of the distribution of the src_bytes feature between the real and synthetically generated data for the U2R attack class.
Explanation: on the next slide

CTGAN results

Explanation: The CDF plot for src_bytes (real vs synthetic data) shows a comparison of the distribution of this feature between the real and synthetically generated data for the U2R attack class.

Interpretation of the CDF plot for src_bytes (real vs synthetic data) plot:

Overall shape of the distributions

Both curves start at 0 and rise to 1, representing the cumulative distribution functions.

The shape of the curves gives insight into how the src_bytes values are distributed in real and synthetic datasets.

Differences between real and synthetic distributions

The real data (blue points) have a rapid increase near zero, indicating many small values in the real data.

The synthetic data increase more gradually at the beginning and spread more evenly across a wider range of values, up to about 6000.

The synthetic points are more uniformly spaced and more numerous, reflecting the larger size of the synthetic set (e.g., 1000 samples vs. around 50 real samples).

CTGAN results

Areas with notable differences

At low values (around 0–500), the real data's CDF rises sharply, showing a concentration of many small src_bytes values, while the synthetic data's CDF rises more slowly.

This suggests the CTGAN generator does not perfectly replicate the high density of small values found in the real data for this feature.

For larger values (>2000), the synthetic data continues to accumulate steadily, while the real data has fewer samples but some extreme large values.

General interpretation

The CTGAN has produced a distribution with a similar overall shape but noticeable local differences in density, especially in the lower value range.

This is common when working with small real datasets (U2R is a rare class).

Further tuning might be needed to better capture these dense regions in the real data.

t-SNE for Evaluating GAN-Based Data Augmentation

Overview

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a **dimensionality-reduction** algorithm

Projects high-dimensional data (e.g., 41-feature NSL-KDD samples) into 2D/3D for visualization.

Preserves local neighborhood structure — similar samples remain close in low-dim space

Role in GAN Data Augmentation

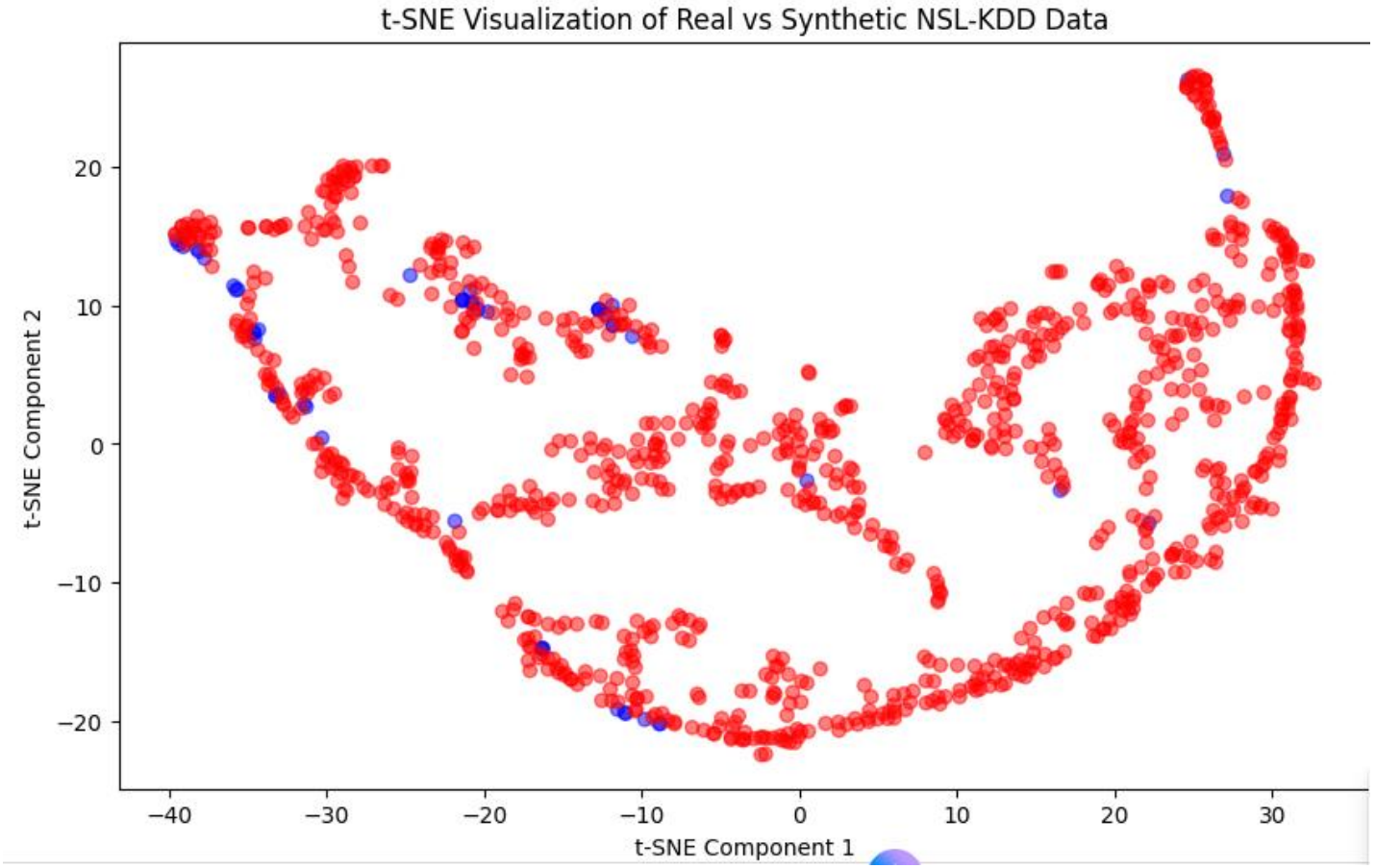
Visualize how **real and synthetic samples overlap** after GAN training.

Detect **mode collapse** (GAN producing limited variations).

Compare data distribution coverage before and after augmentation.

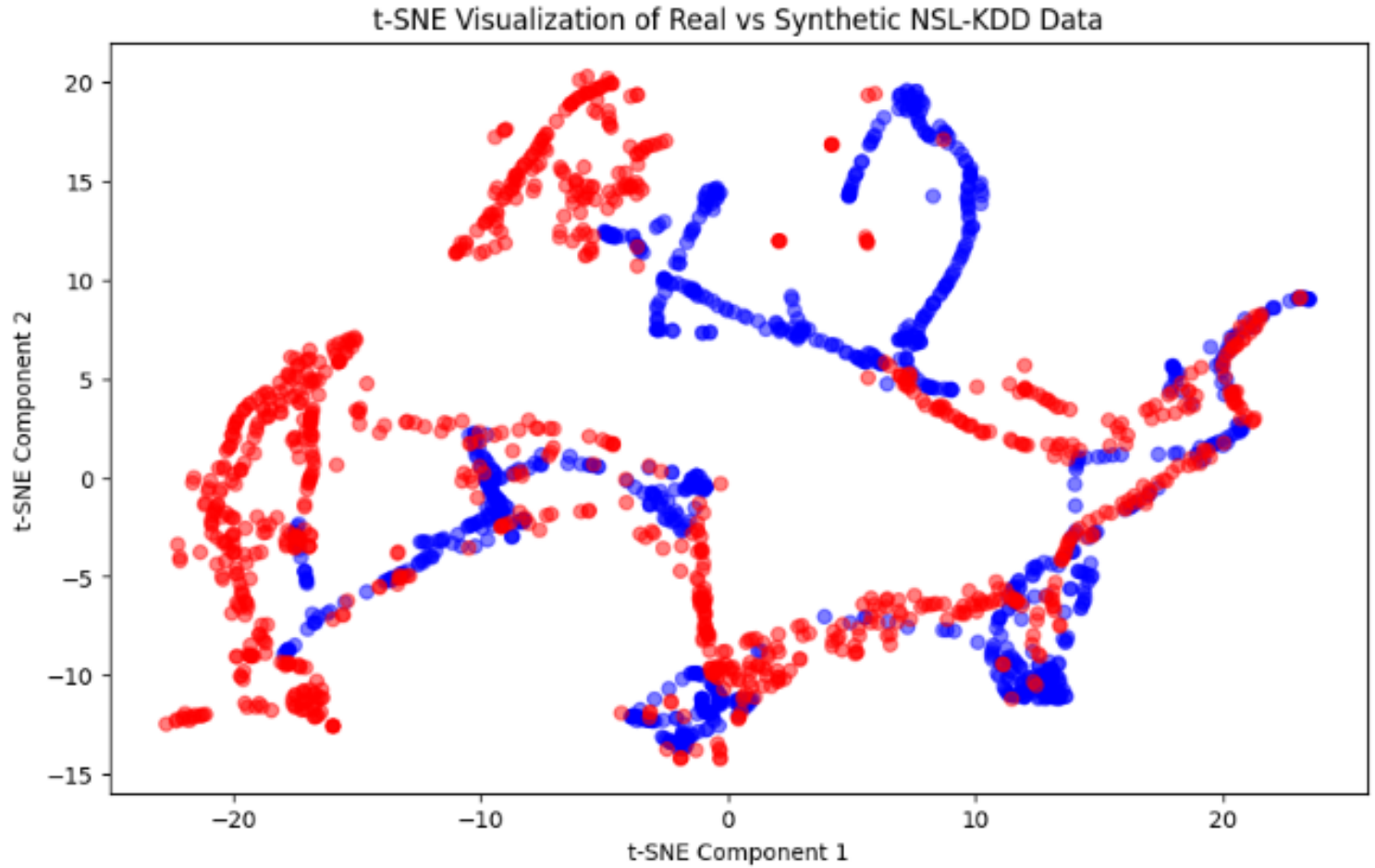
Explore latent space structure to design conditional GANs.

CTGAN results, U2R data



Blue: real data (52 samples) / Red: synthetic data

CTGAN results, U2R data



Blue: real data (1000 samples, SMOTE) / Red: synthetic data

Conclusions

- Discriminative Models (RF, SVN, XGBoost, etc.) can be used to implement IDS to detect primarily known attacks
- Generative models can be used to protect against Zero-Days attacks (by learning the normal traffic behaviour) and to augment misrepresented classes in training datasets
- Performance measurement and explainable AI must be added

References

Original GAN Paper by Ian Goodfellow : https://arxiv.org/pdf/1406.2661

Applying Generative Machine Learning to Intrusion Detection: A Systematic Mapping Study and Review https://dl.acm.org/doi/10.1145/3659575

IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection https://arxiv.org/pdf/1809.02077

Development of a Machine-Learning Intrusion Detection System and Testing of Its Performance Using a Generative Adversarial Network https://www.mdpi.com/1424-8220/23/3/1315

Synthetic Network Traffic Data Generation: A Comparative Study https://arxiv.org/pdf/2410.16326v2

A Review of Tabular Data Synthesis Using GANs on an IDS Dataset https://www.mdpi.com/1270532