WICHITA STATE UNIVERSITY

Detecting botnets hidden in DNS over HTTPS traffic: An Autoencoder approach

by:
Aman Kumar Gupta
X397J446
EECS Department

Advisor:
Dr. Sergio A. Salinas Monroy
EECS Department

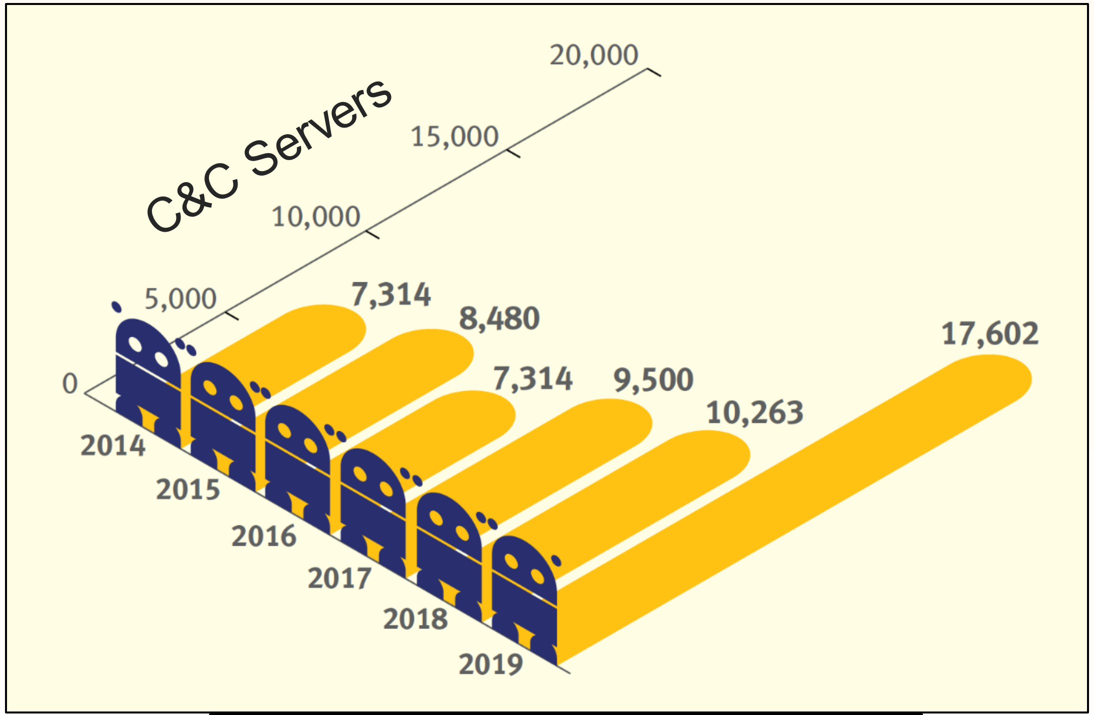
Contents

- General statistics
- Botnet Architecture
- Previous works on DGA Detection
- DoH traffic detection using Autoencoders
- Previous works on DoH detection
- Our Proposed Solution
- Future Works
- Conclusion

Wichita State University

Number of infected devices has increased continuously over the previous years

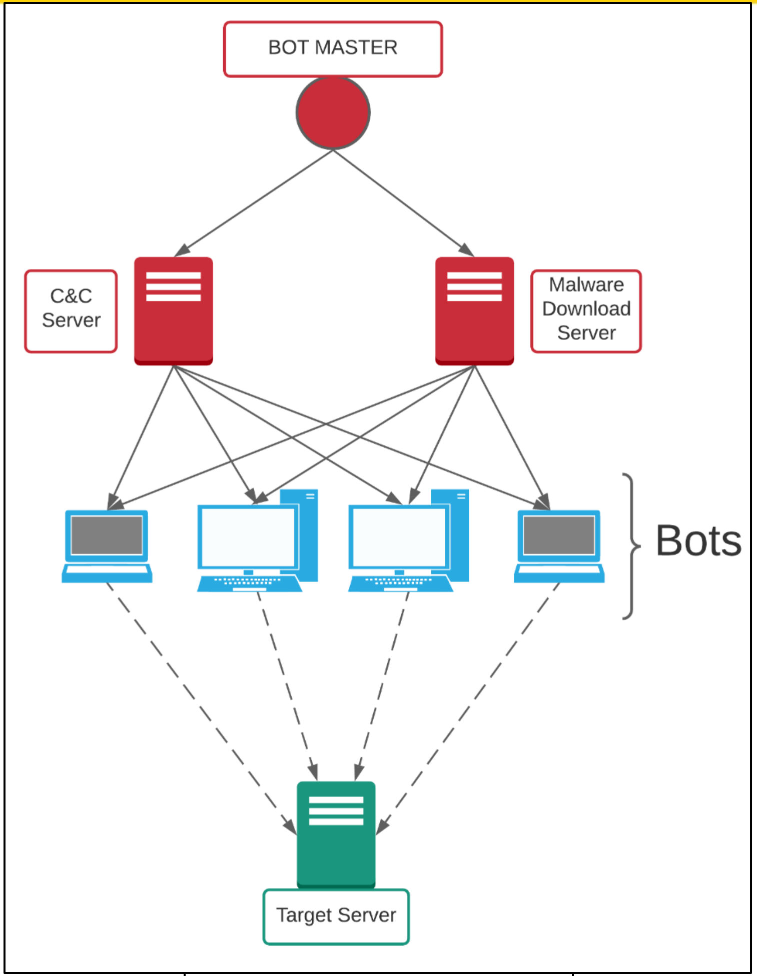
- 7.7 million IoT devices are connected to the Internet
- Only 1 out of 20 are secured.
- Mirai infected 300,000 devices by 2016.
- 913% increase in the number of Emotet samples in 2019.
- 17,602 C&C servers were discovered in 2019



C&C Servers identified over the years

Botnet Architecture

- Bots
- Botmaster
- Binary-download server
- Command and Control server (C&C server)
- Target
- What does a Botnet Attack look like?



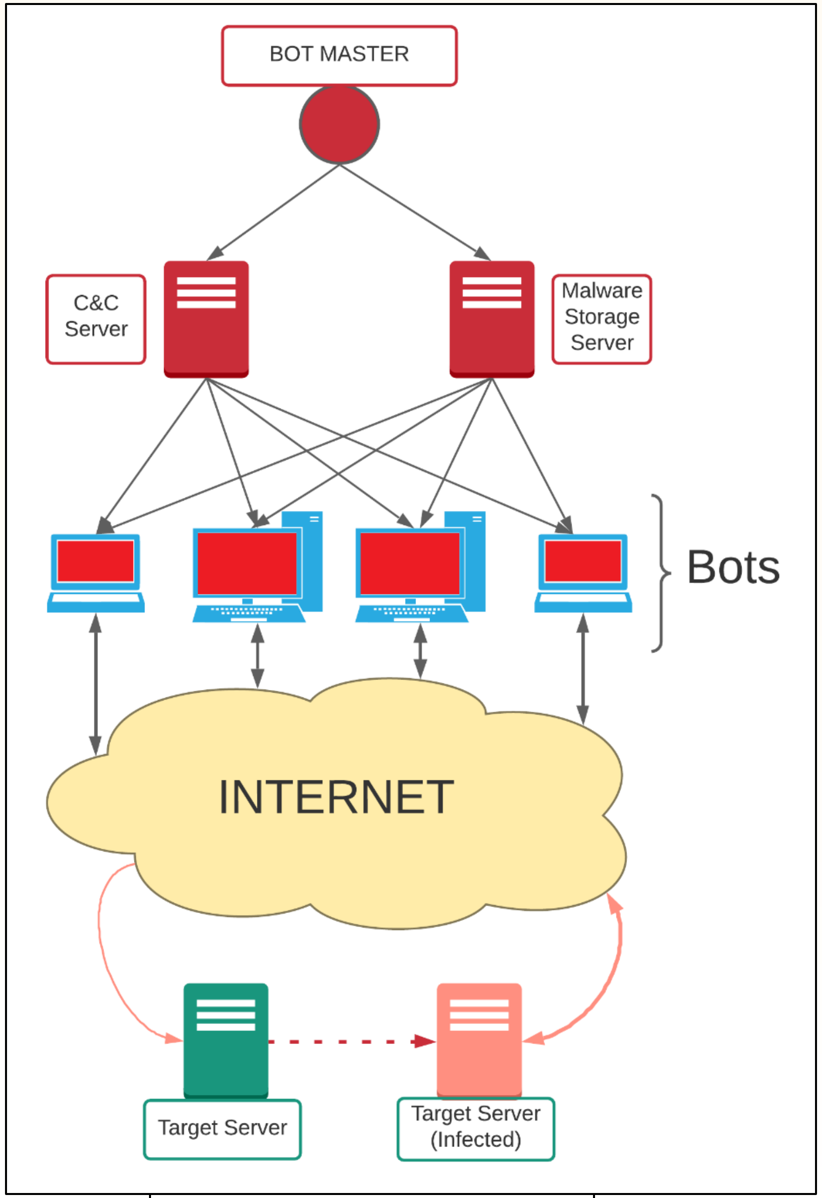
Botnet Architecture

Bot Recruitment

- Botmaster spreads the botnet malware

- Bots download malware from server

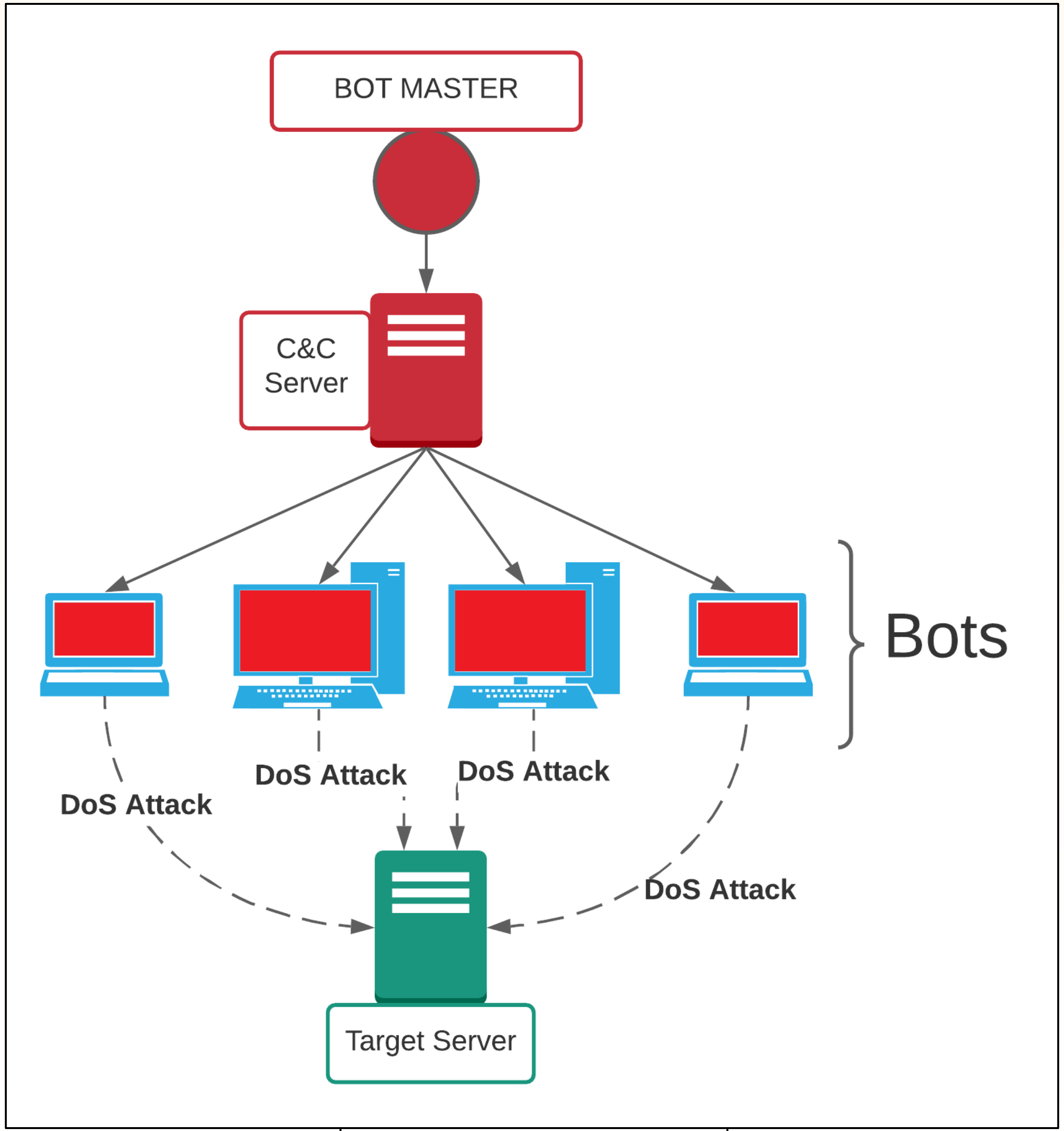
- Bots report status back to the C&C server.



Botnet Infection phase

Botnet Attacks

- Botmaster selects a target.
- Botmaster sends instructions to bots via the C&C server
- Bots execute the instructions.
- Target unable to provide service to actual users.



Botnet Attack phase

Communications between bots and C&C server

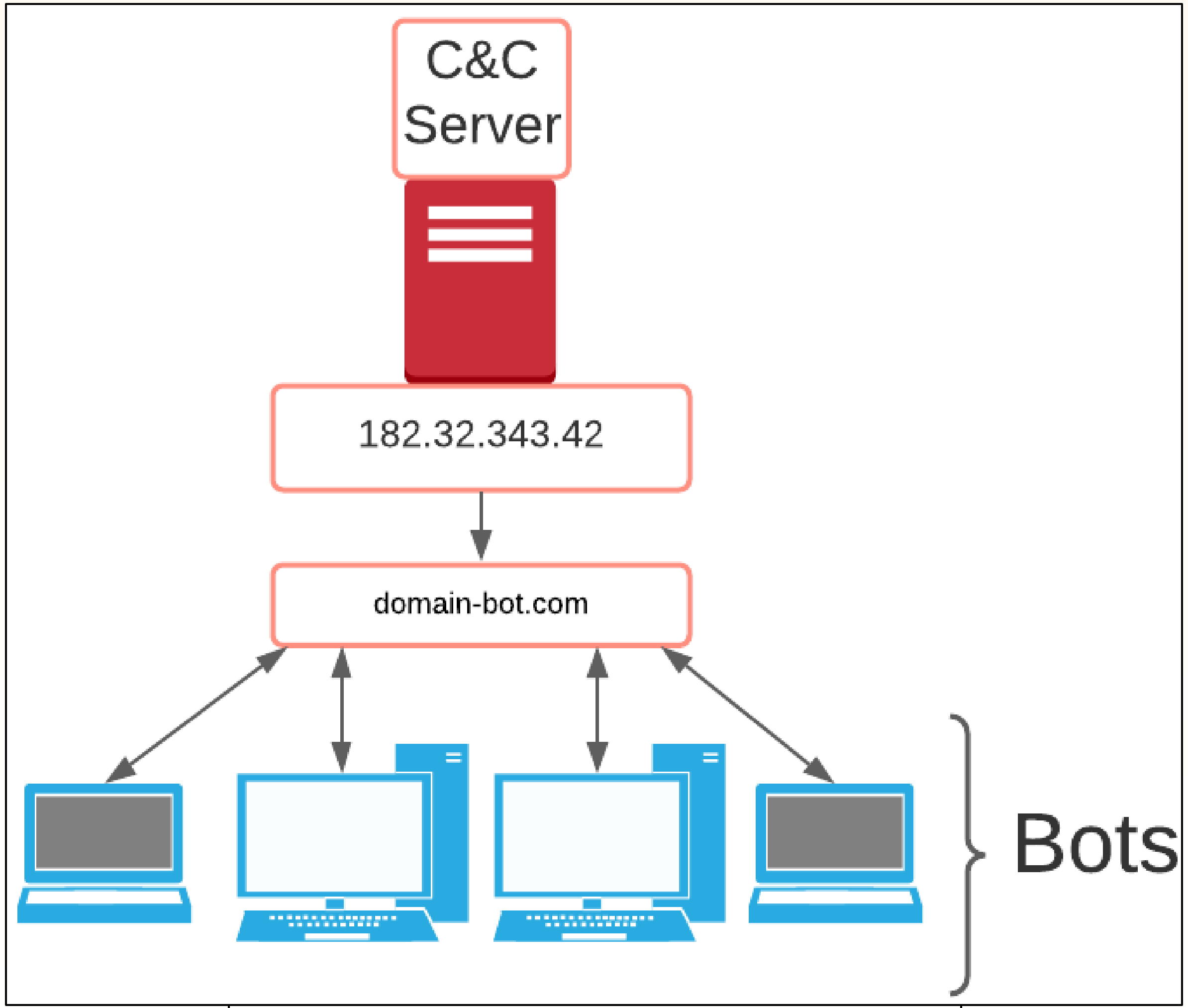
- Methods of communication between C&C and the bots
  - Single static IP/domain
  - List of static IPs/domains
  - Pseudo-Randomly generated domains (DGAs)

C&C Server running on a static IP/domain

- Domain hardcoded in the malware code

- Single point of failure

- Easily to detect



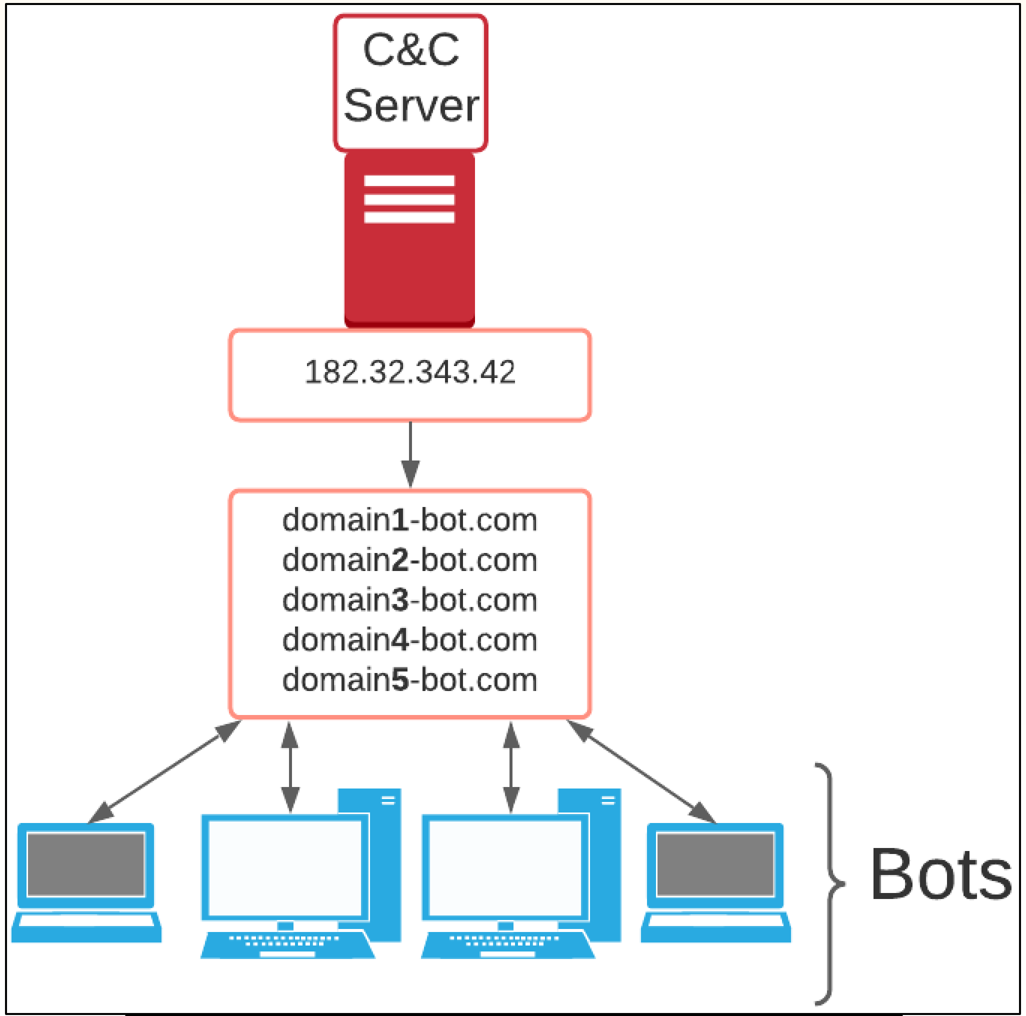
C&C Server running on a static IP/domain

C&C Server running on a list of IPs/domains

- Domain list hardcoded in the malware code

- Single point of failure

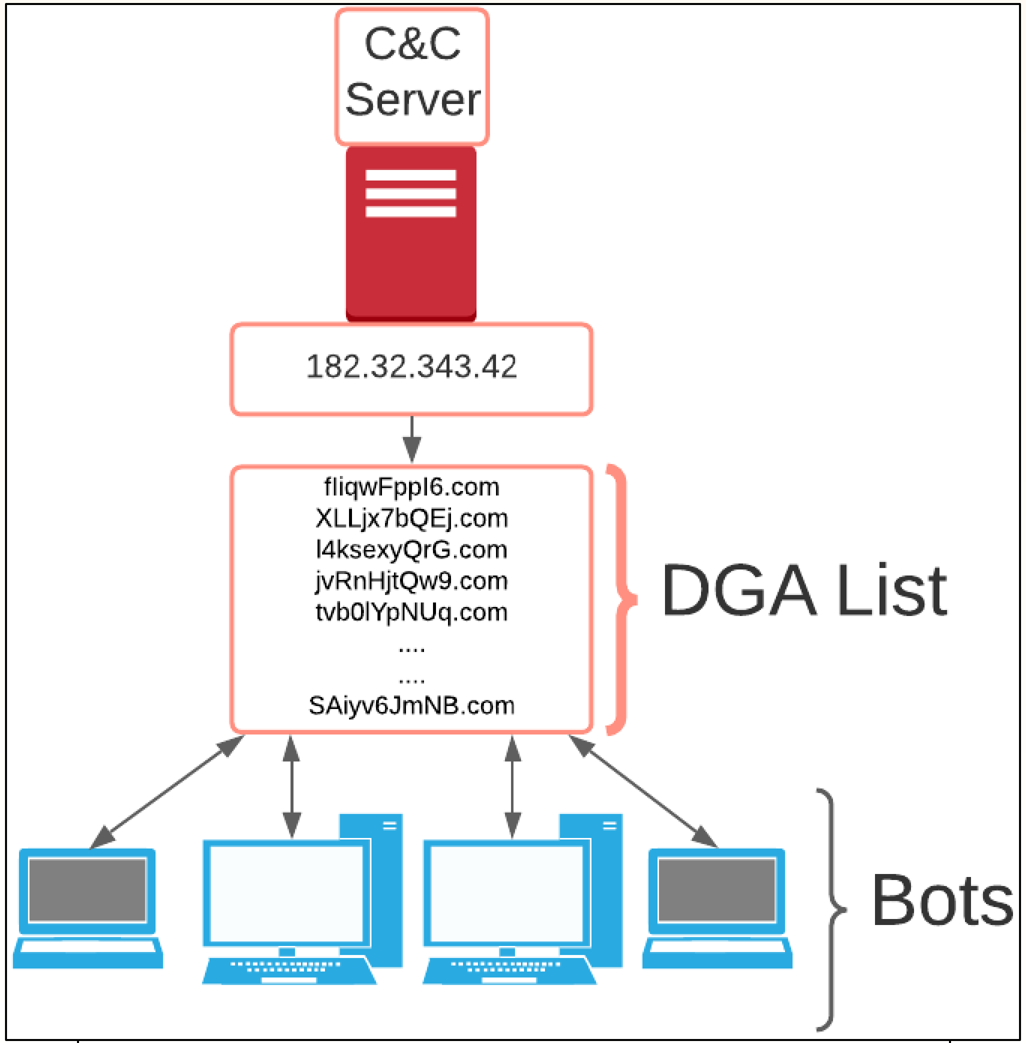
- Easily to detect and shutdown



C&C Server running on a list of IPs/domains

C&C Server running on Pseudo-Random domains

- Domain Generation Algorithm (DGA) built into the malware code
  - Same seed as the C&C server.
- Hard to detect
- Requires reverse engineering the malware and DGA



C&C Server running on Pseudo-Random domains

How can we detect botnets?

- C&C's domain identified by DNS lookups performed by the bots.

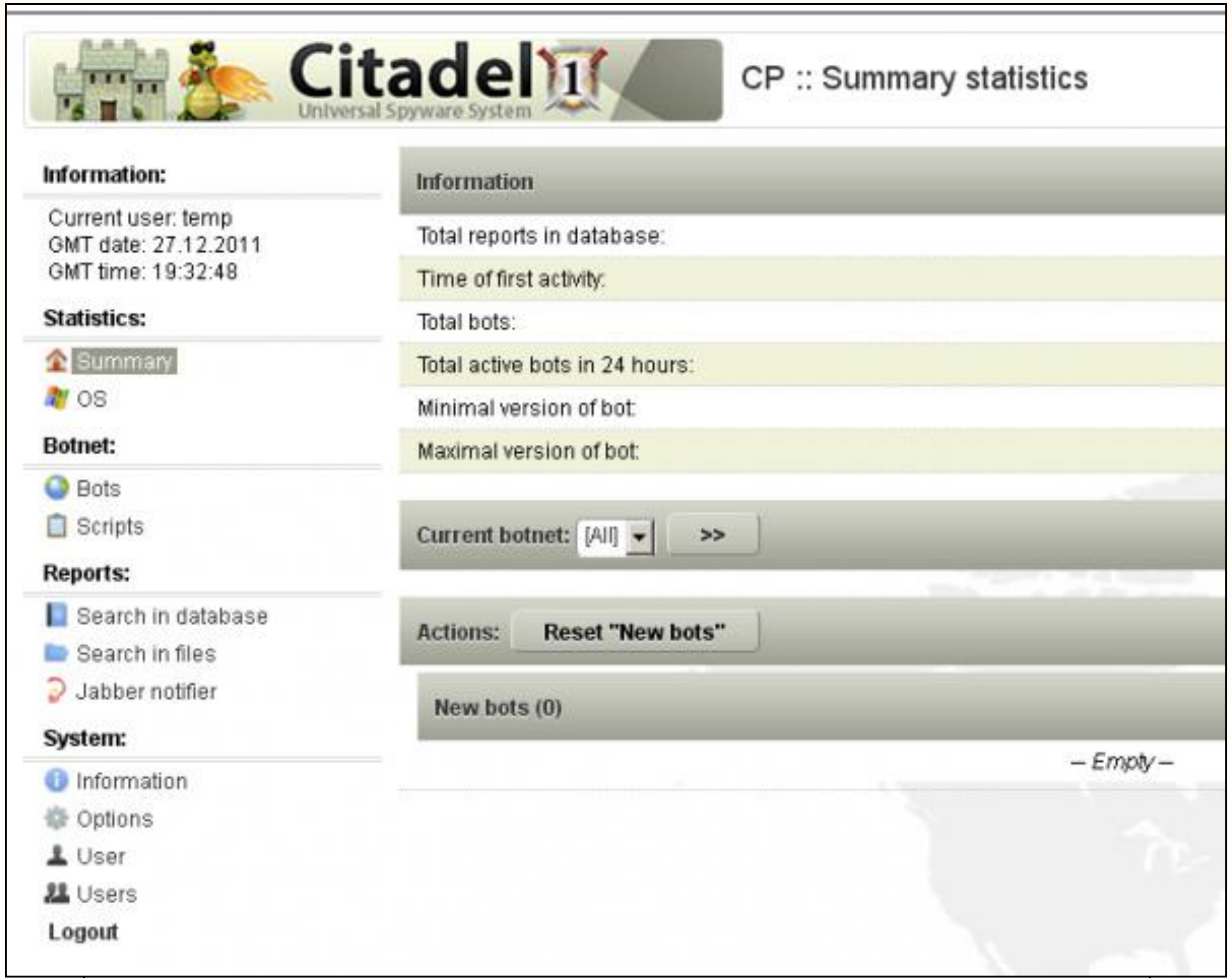
- DNS server replies with an NX domain response for unregistered domains.

- NX domains can be identified by checking the type of DNS reply.

- Key is to detect DGA URLs, or DNS queries for those URLs and block them.

Real world botnet take downs

- **Citadel**
  - 1000 domains seized
  - 11 million victim computers
  - Cause of a $500 million loss
  - Taken down June 2013
- **ZeroAccess**
  - 2 million victim computers
  - Cause of a $2.7 million loss
  - Taken down Dec 2013
- **Necurs**
  - Identified domains 25 months in the future
  - Blacklisted domains



CP :: Summary statistics

Information:

Current user: temp
GMT date: 27.12.2011
GMT time: 19:32:48

Information

Total reports in database:

Time of first activity:

Total bots:

Total active bots in 24 hours:

Minimal version of bot:

Maximal version of bot:

Current botnet: [All]

Actions:

New bots (0)

– Empty –

A screenshot of the Web-based Citadel botnet control panel.

Previous works on **DGA Detection**

- Mac, et al. **DGA botnet detection using supervised learning methods.** *Proceedings of the Eighth International Symposium on Information and Communication Technology. 2017.*
  - *Uses Hidden Markov Models, LSTMs, and Support Vector Machines*
- Woodbridge, et al. **Predicting domain generation algorithms with long short-term memory networks**. *arXiv preprint arXiv:1611.00791 (2016).*
  - *Ability to detect DGA and botnet families from the URL*
- Tran, et al. **A LSTM based framework for handling multiclass imbalance in DGA botnet detection**. *Neurocomputing 275 (2018): 2401-2413.*
  - *Ability to detect DGA and botnet families from the URL*
  - *Improves upon the previous paper's class imbalance*
- Sidi, et al. **Helix: DGA Domain Embeddings for Tracking and Exploring Botnets**. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management. 2020.*
  - *Uses Autoencoder*
  - *Capable of detecting DGAs and botnet families.*
  - *Also, able to track botnet campaigns across network data*
  - *Currently in use by a major ISP*

Preliminary work by A. Gupta

Malicious URL detection (Mal-U-Detect)

- LSTM & CNN architectures to detect malicious DGAs.
- Improved upon the **Predicting Domain Generation Algorithms with Long Short-Term Memory Networks** experiment by Woodbridge et al.
- Training time for CNN model over 50 epochs is ~60 minutes
  - Dataset size 1.3 million records
- 98.2% accuracy.

DoH traffic detection using Autoencoders

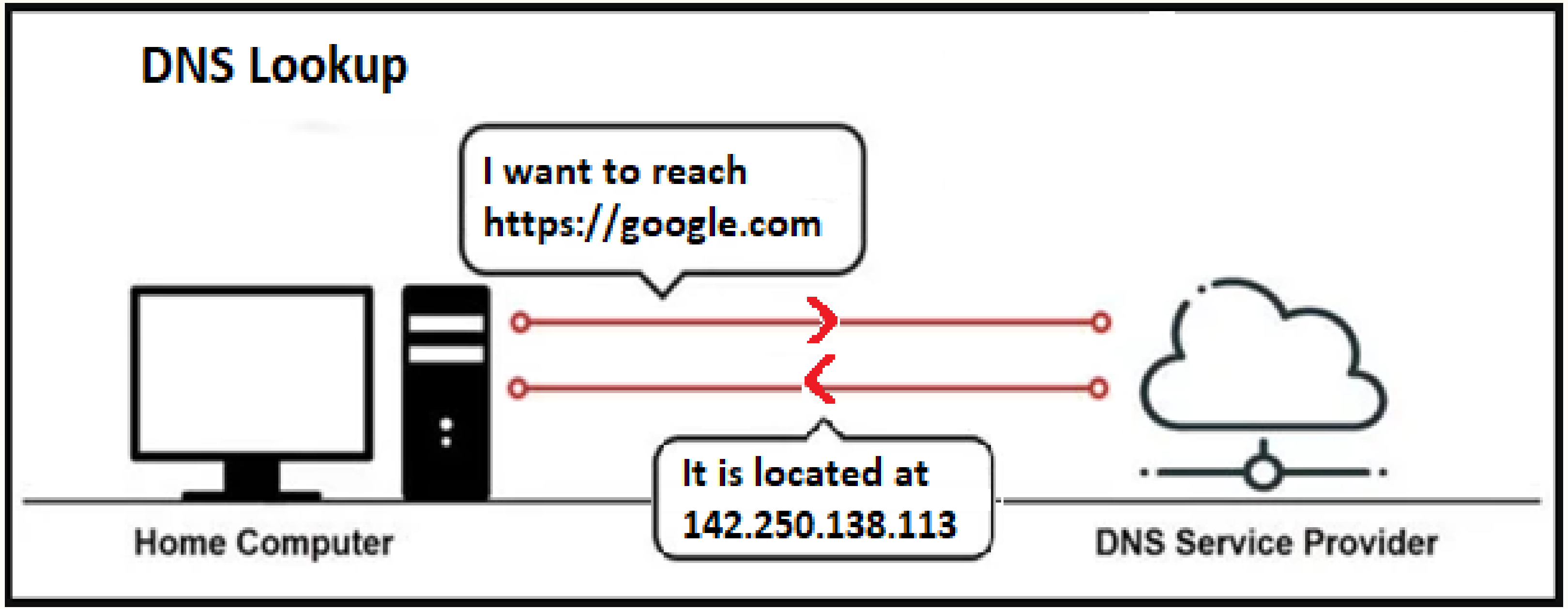
DNS over HTTPS makes it impossible to observe the NXDOMAIN responses.

- DNS
- HTTPS
- DNS over HTTPS

DNS

- Domain Name System

- DNS lookup

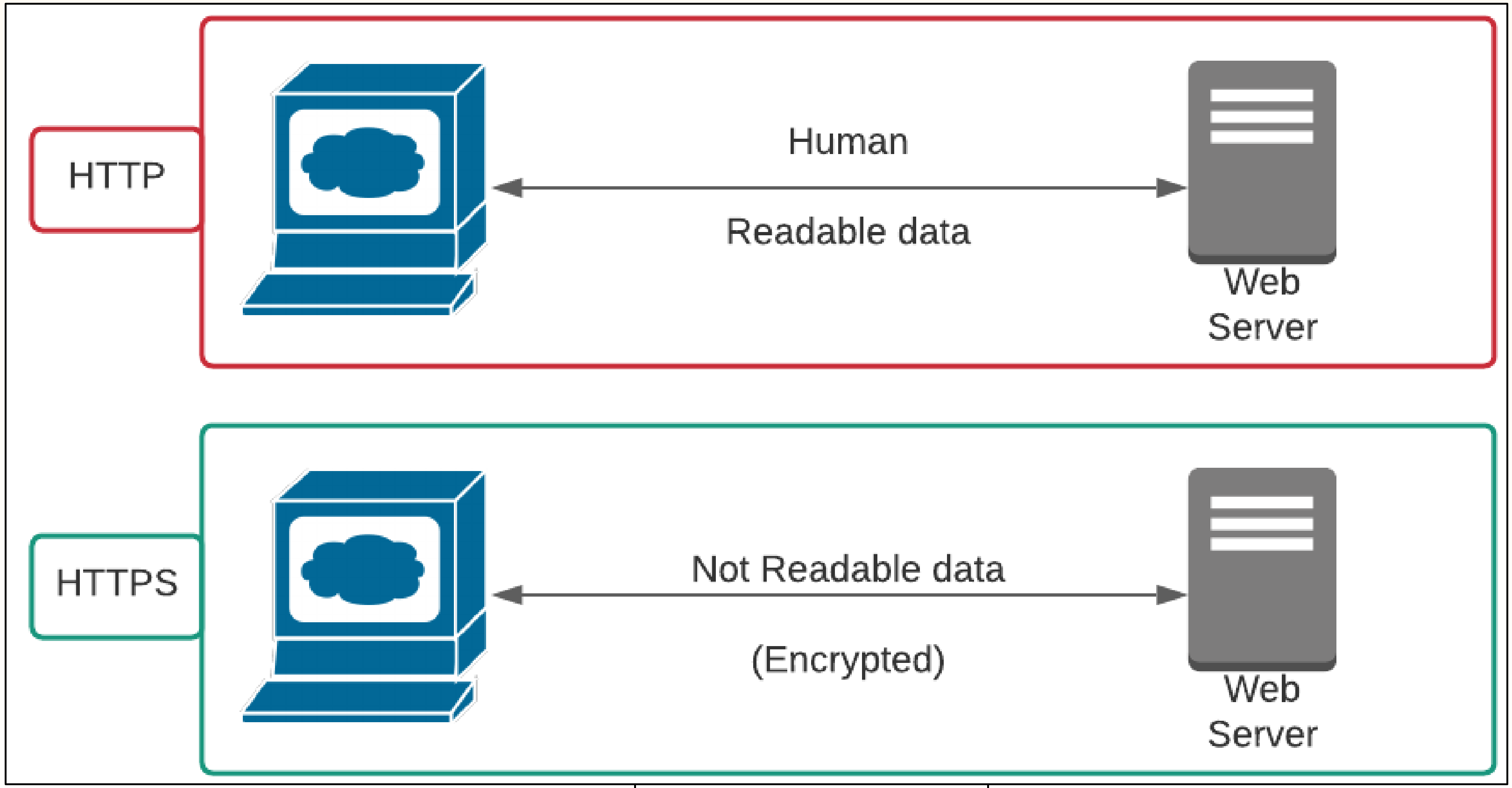

DNS Lookup
I want to reach https://google.com
It is located at 142.250.138.113
Home Computer
DNS Service Provider

DNS Lookup

HTTPS

How does HTTPS work?

- TLS handshake between the browser (**client**) and the server.
- After handshake, secure HTTPS connection established.

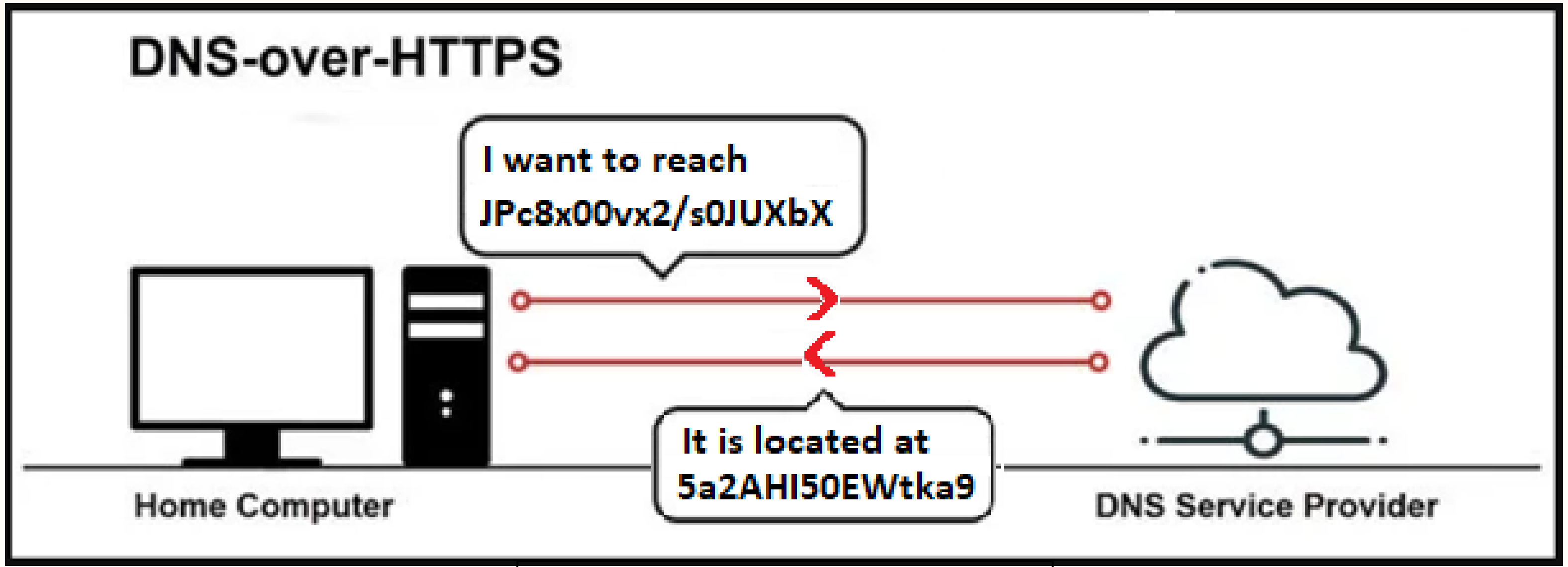


HTTP vs HTTPS

DNS over HTTPS

What is DoH?

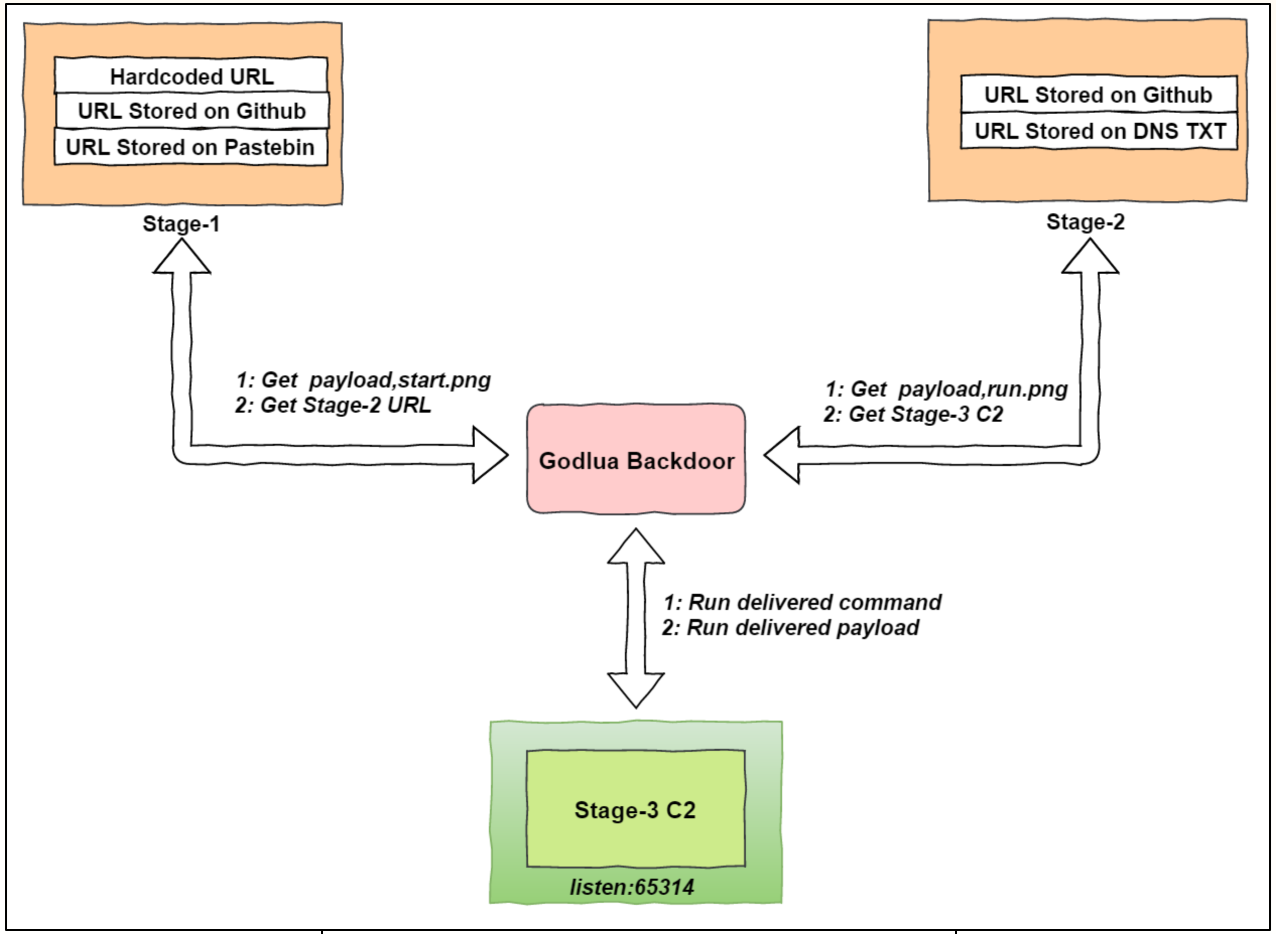
- DNS over HTTPS encrypts DNS traffic


DNS-over-HTTPS
I want to reach JPc8x00vx2/s0JUXbX
It is located at 5a2AHI50EWtka9
Home Computer
DNS Service Provider

DNS over HTTPS procedure

Real world example of botnet using DoH

- First Botnet detected Godlua's Linux variant detected using DoH for infection

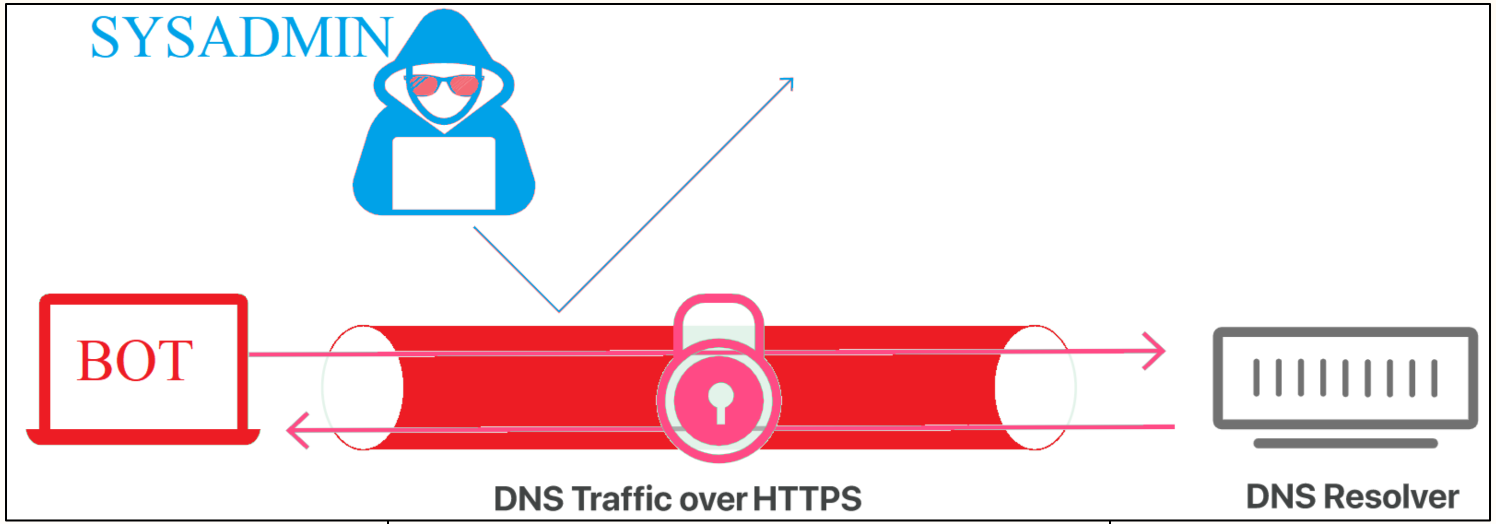


Godlua botnet execution process

Botnets using DoH to avoid detection

- SysAdmin only observes encrypted data.

- New ML solutions would need to rely on network traffic characteristics (explain)
  - Packet length
  - Bytes sent out
  - Average packet size, etc., …



DoH blocking SysAdmin’s monitoring

Previous works on DoH detection

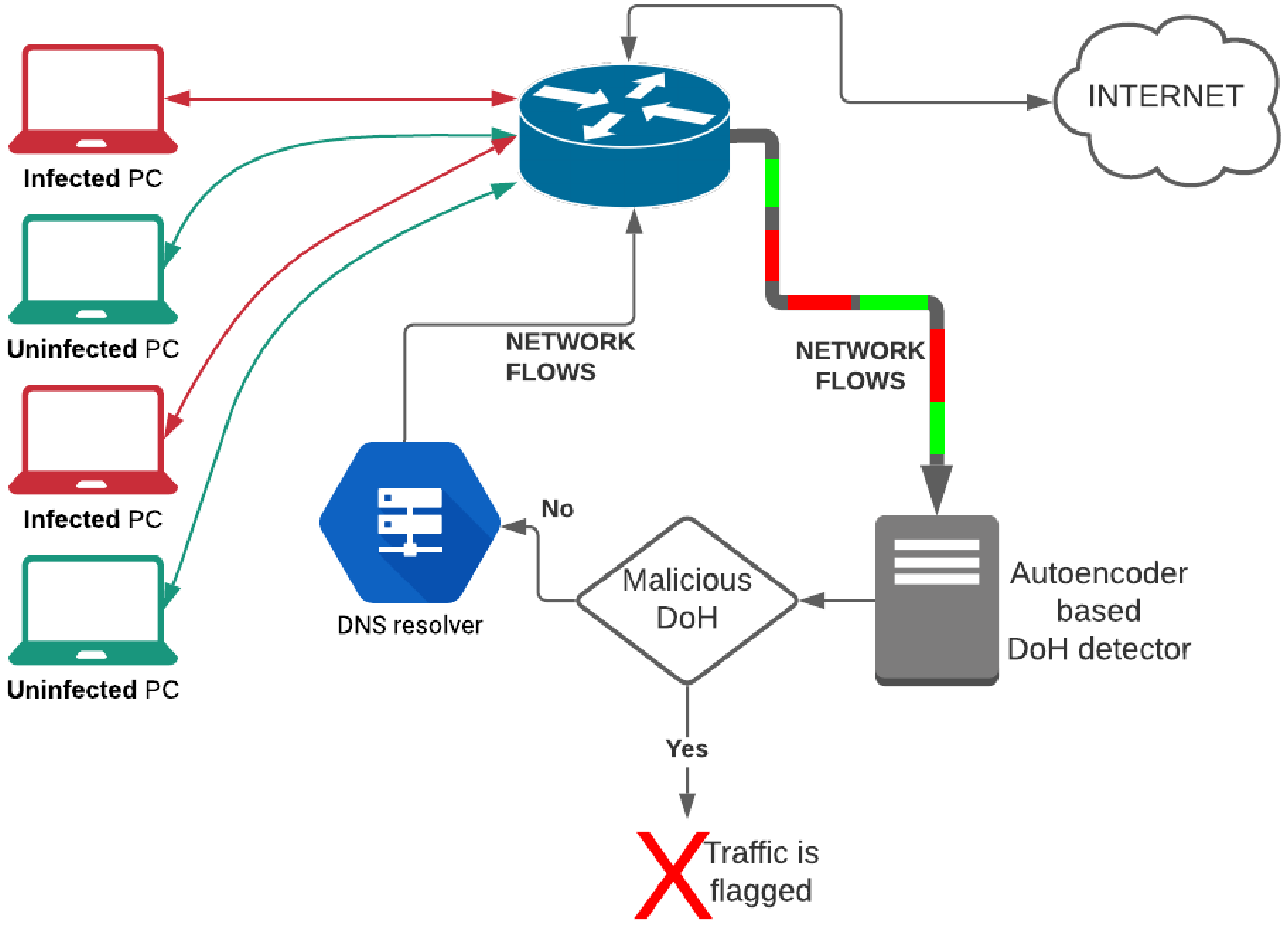
- No existing works on detecting DGAs over DoH

- Vekshin, Dmitrii and et al. **Doh insight: Detecting dns over https by machine learning**.
  - Created a dataset with 940882 records
  - The authors successfully classified HTTPS and DoH traffic:
    - K-Nearest Neighbors
    - Decision Tree
    - Random Forest
    - Naïve Bayes
    - Ada-boosted Decision Tree
  - They were able to achieve 99.6% classification accuracy.

Our proposed solution

- We propose to use Autoencoder for DGA detection
  - Unsupervised Learning

- Attempt DoH vs HTTPS classification since DGA datasets don't exist.

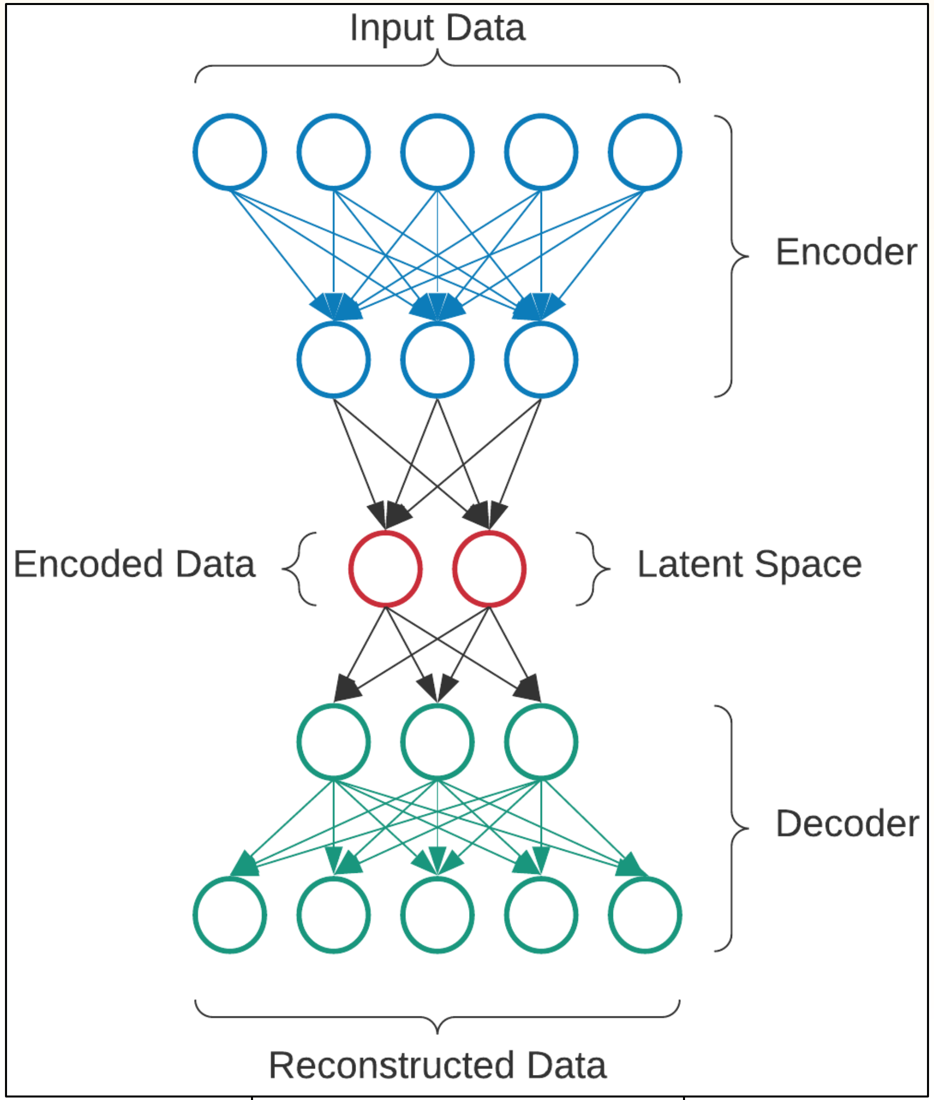


Autoencoders

- Neural Network trained to produce copy of input.

- Hidden layer encodes the input to a smaller encoding called **code.**

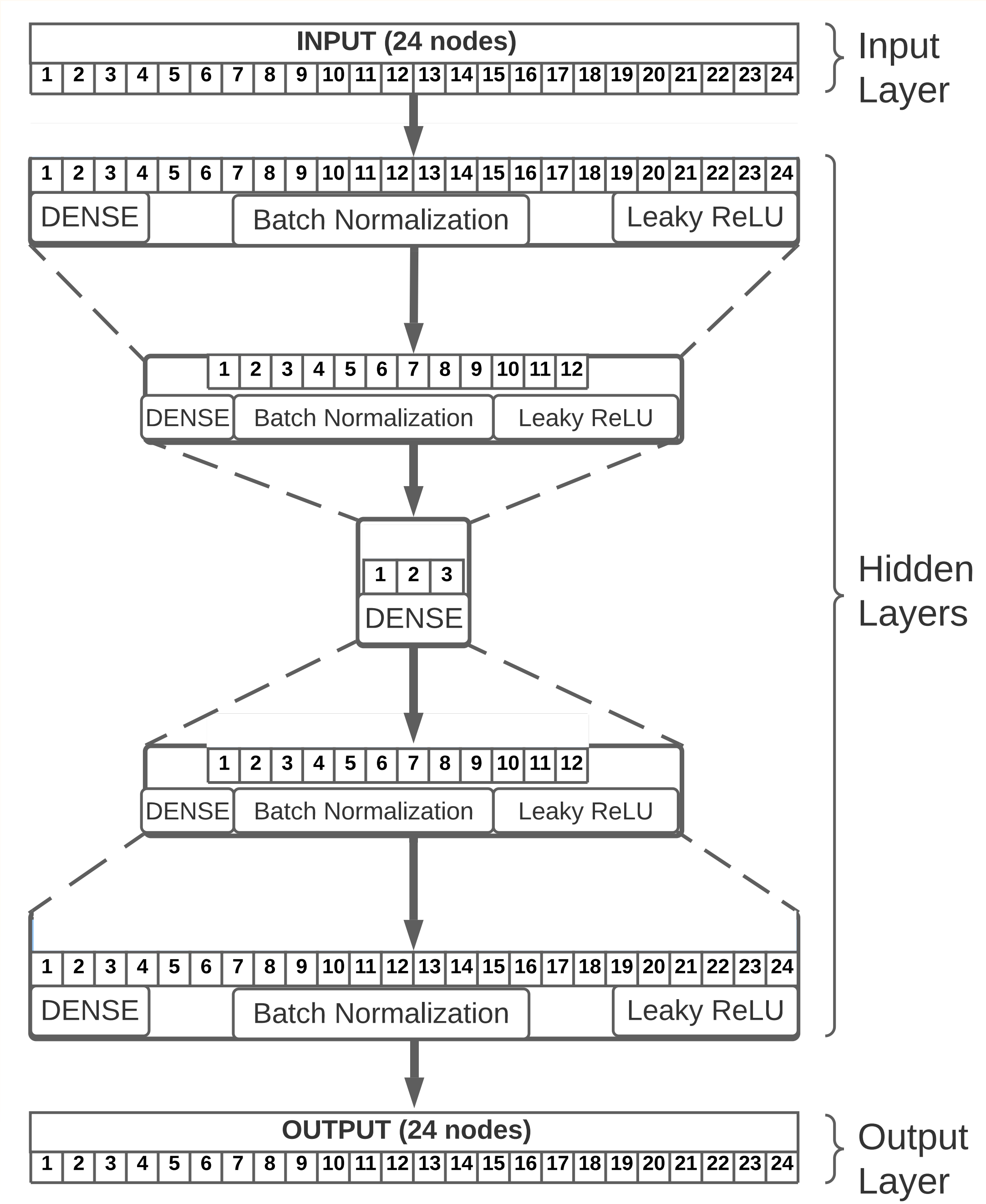
- Encodings are used to recreated the input data.



Autoencoder example

How does this project use an Autoencoder model?

- 5 hidden layers
    - Each encoder and decoder has:
        - Dense function
        - Batch Normalization
        - Leaky ReLU function
- Inputs
    - 24 variables (time, av_pkt_size_in, av_pkt_size_out, etc.…)
- Encoder Output
    - 3 variables *"code"*
- KMeans used for clustering.



Autoencoder Model

Hardware/Software architecture

- Google Colab research environment
  - 2 Intel(R) Xeon(R) CPUs
    - Clocked at 2.30 GHz
  - 13 GBs RAM

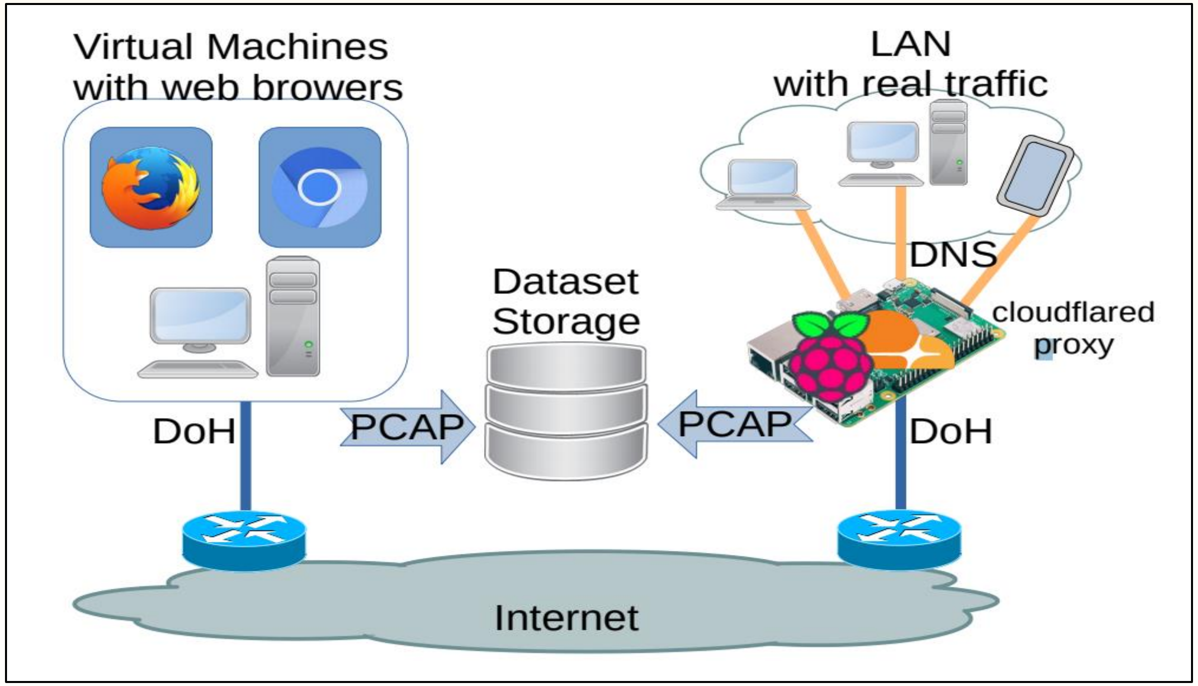
- Running code in a Python notebook
  - Epochs: 50
  - Batch size: 32

- Average run time per epoch is 54 seconds

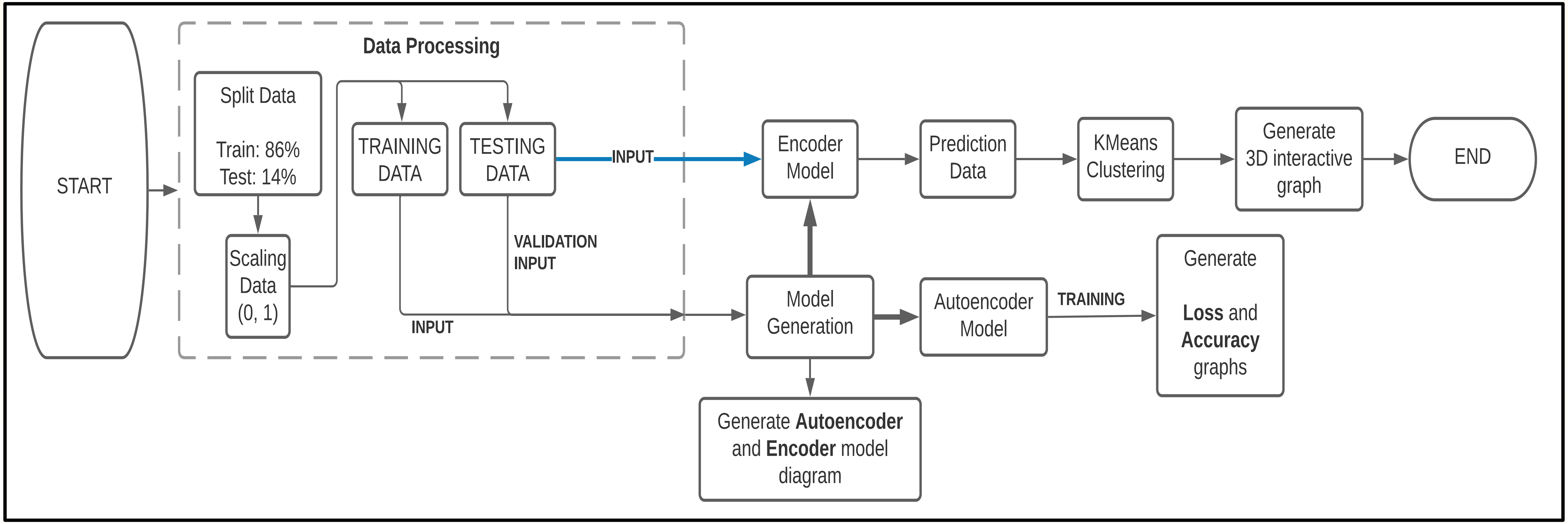
Data set being used

- From **Doh insight: Detecting dns over https by machine learning** experiment

- Extracted network flow characteristics from PCAP files
    - Bytes outgoing
    - Bytes incoming
    - Packets outgoing
    - Packets incoming
    - Packet bursts …



Full architecture



Data Processing

Split Data

Train: 86%
Test: 14%

Scaling Data (0, 1)

TRAINING DATA

TESTING DATA

INPUT

VALIDATION INPUT

INPUT

START

Encoder Model

Prediction Data

KMeans Clustering

Generate 3D interactive graph

END

Model Generation

Autoencoder Model

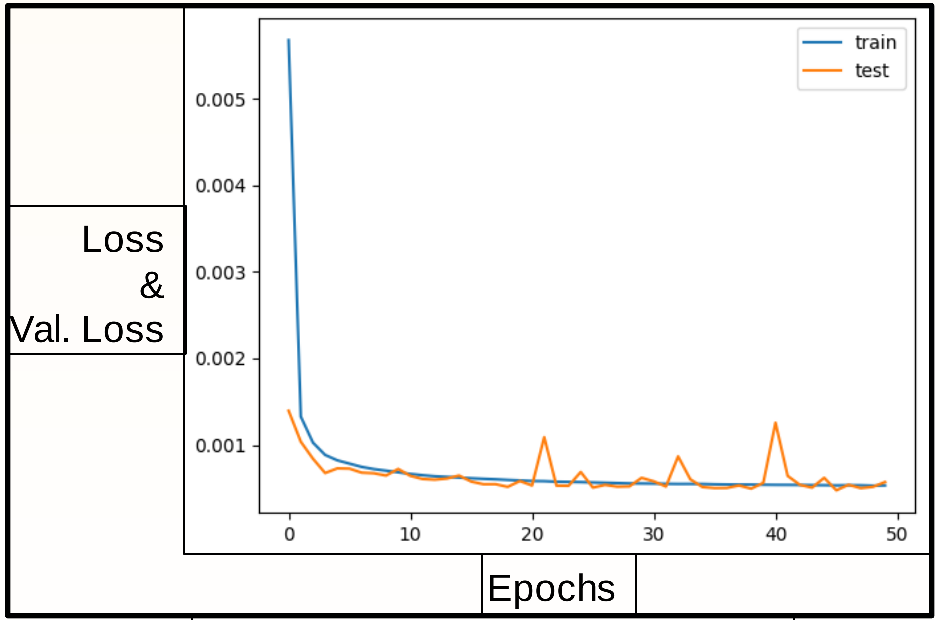
TRAINING

Generate **Loss** and **Accuracy** graphs

Generate **Autoencoder** and **Encoder** model diagram

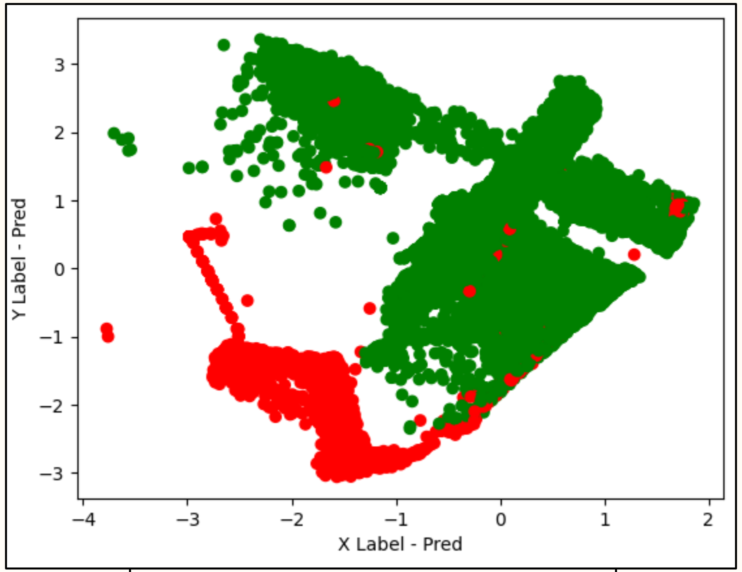
Initial Results

- We tested different variations for the Autoencoder
- 2 Neurons & 3 Loss Functions
- Mean Squared Error
    - Accuracy of autoencoder model: 91.79%

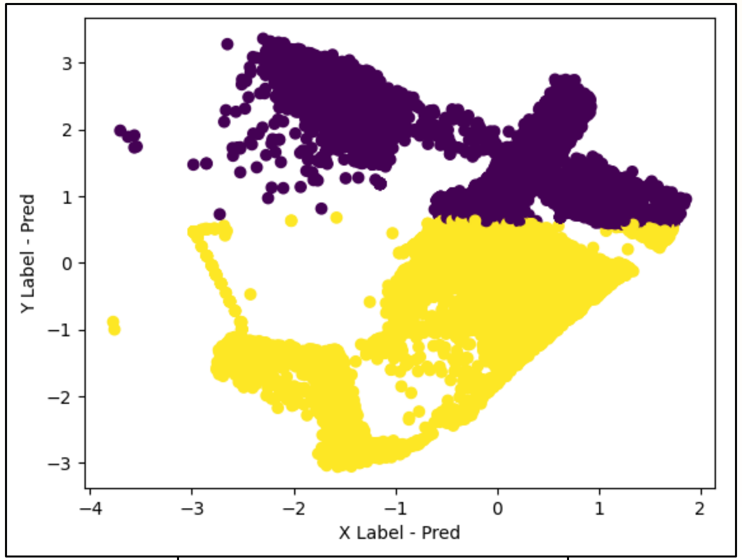


Loss/Val. Loss VS Epochs

N=131724	Pred DoH	Pred HTTPS
Actual DoH	4777	310
Actual HTTPS	36429	90208



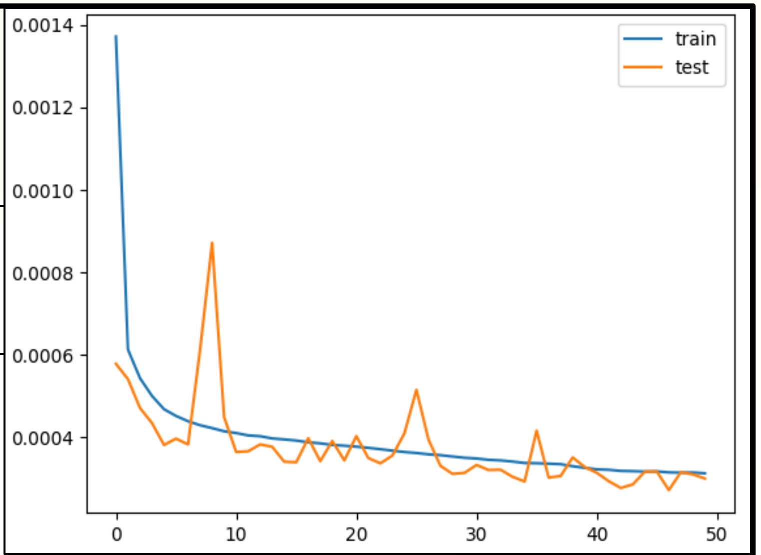
Clustering – Ground Truth



Clustering – KMeans

Initial Results

- Mean Squared Logarithmic Error (2 Neurons)
- Accuracy of Autoencoder model: 90.84%

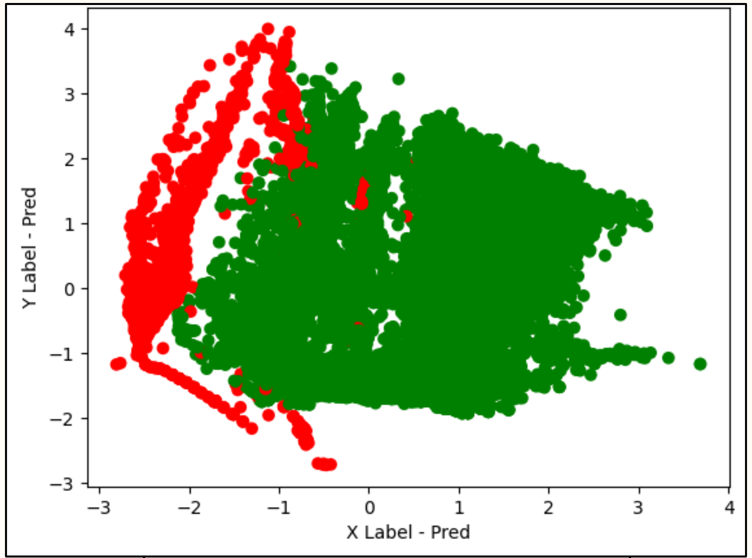


Loss & Val. Loss

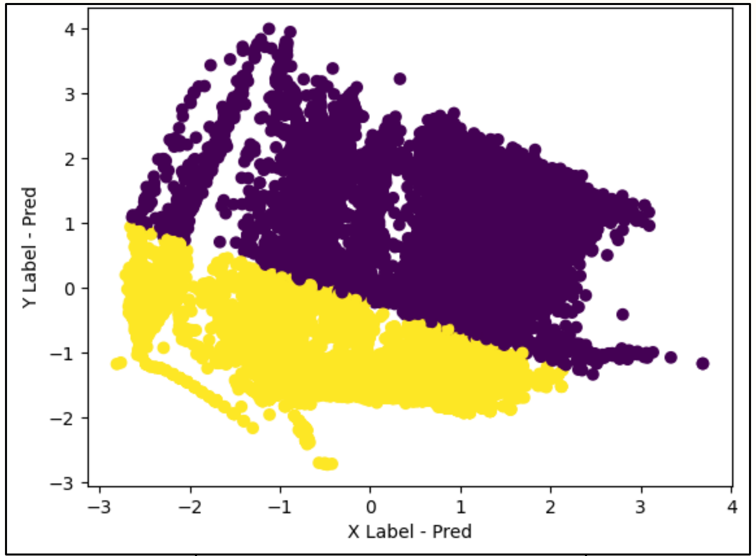
Epochs

Loss/Val.Loss VS Epochs

N=131724	Pred DoH	Pred HTTPS
Actual DoH	4561	107752
Actual HTTPS	526	18885



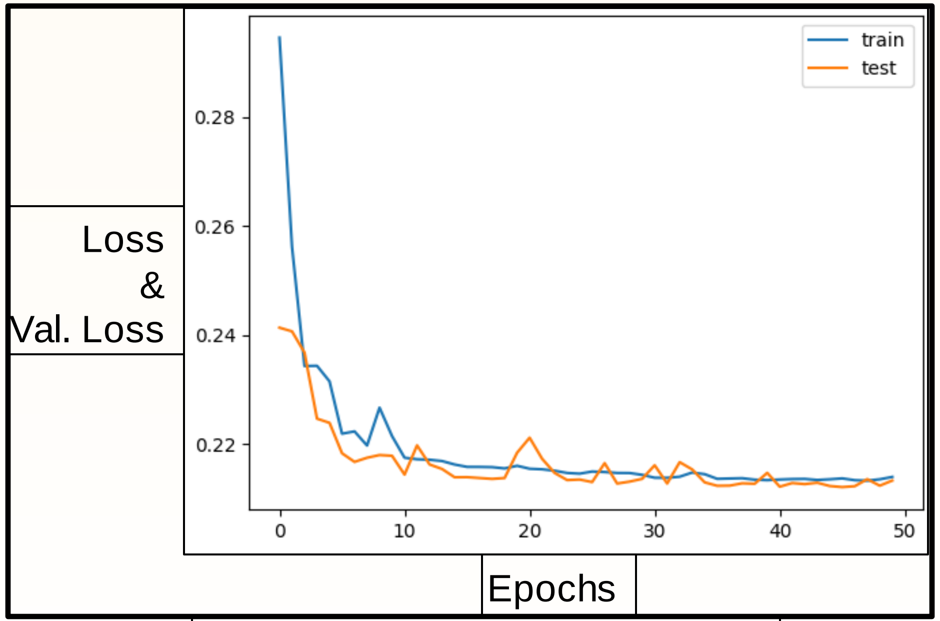
Clustering – Ground Truth



Clustering – KMeans

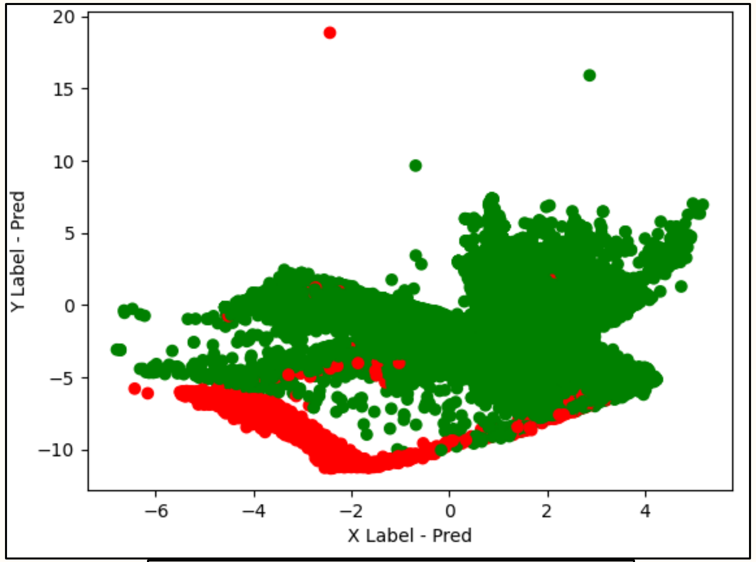
Initial Results

- Binary Cross Entropy (2 Neurons)
- Accuracy of Autoencoder model: 83.14%

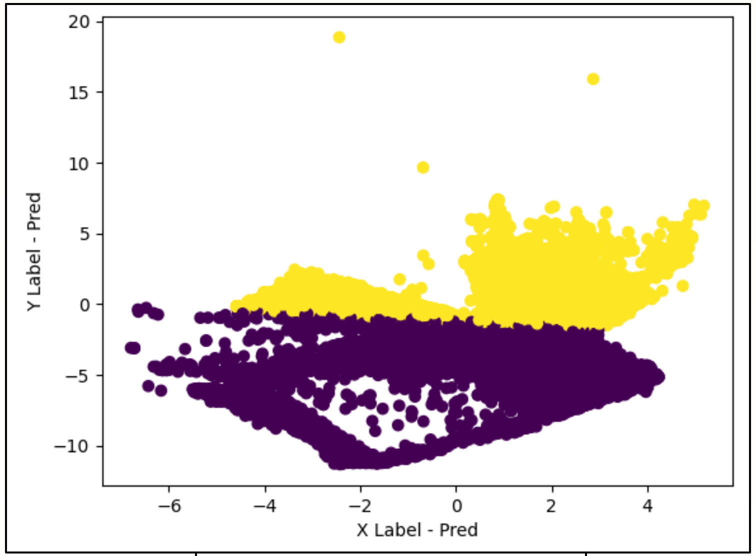


Loss/Val.Loss VS Epochs

N=131724	Pred DoH	Pred HTTPS
Actual DoH	321	4766
Actual HTTPS	98980	27657



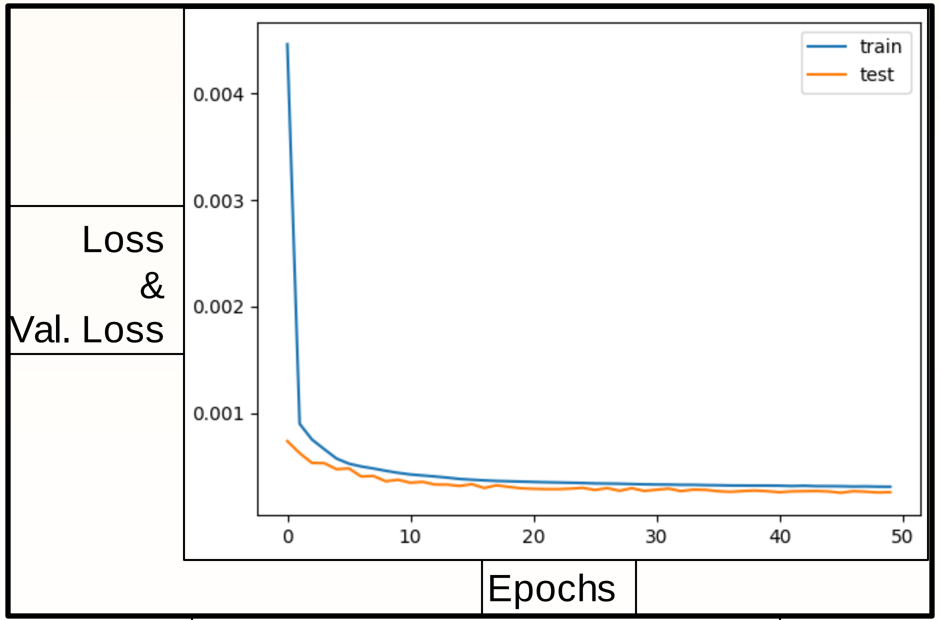
Clustering – Ground Truth



Clustering – KMeans

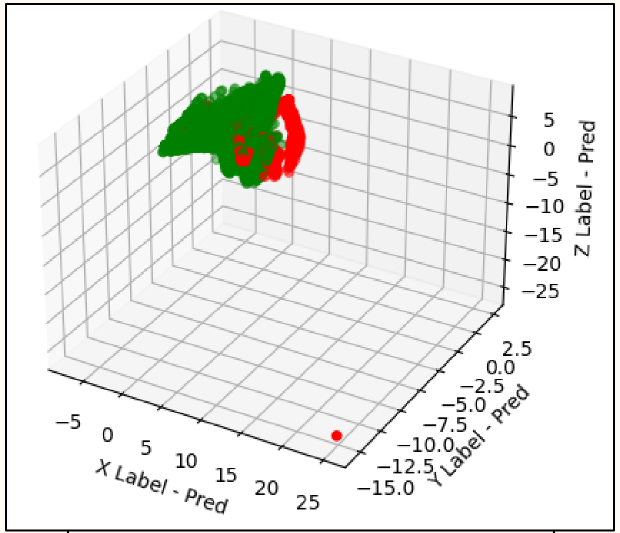
Initial Results

- 3 Neurons & 3 Loss Functions
- Mean Squared Error
    - Accuracy of autoencoder model: 91.79%

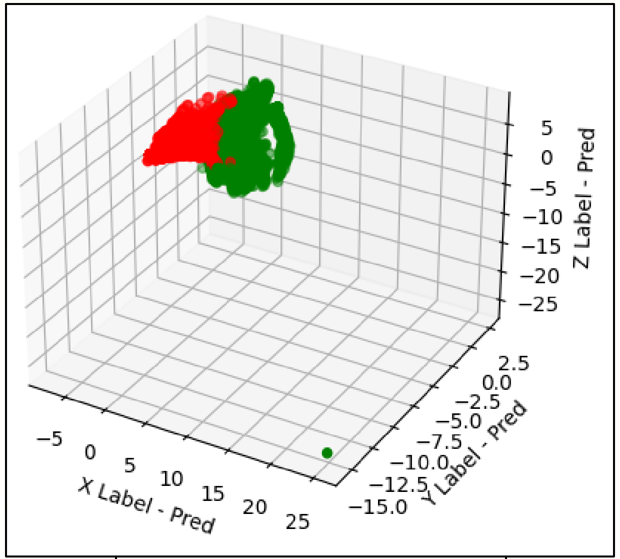


Loss/Val.Loss VS Epochs

N=131724	Pred DoH	Pred HTTPS
Actual DoH	123	4964
Actual HTTPS	18954	107683



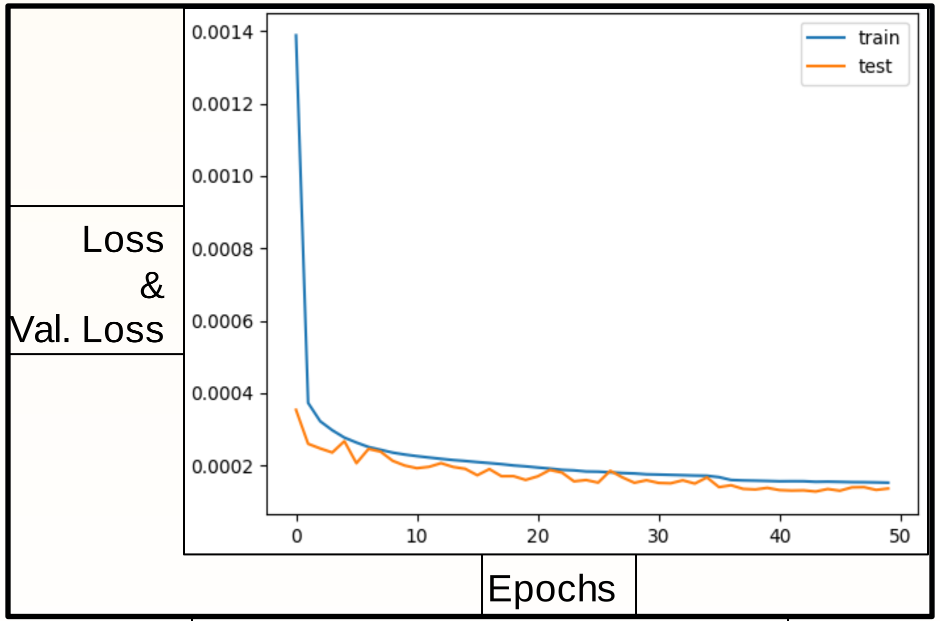
Clustering – Ground Truth



Clustering – KMeans

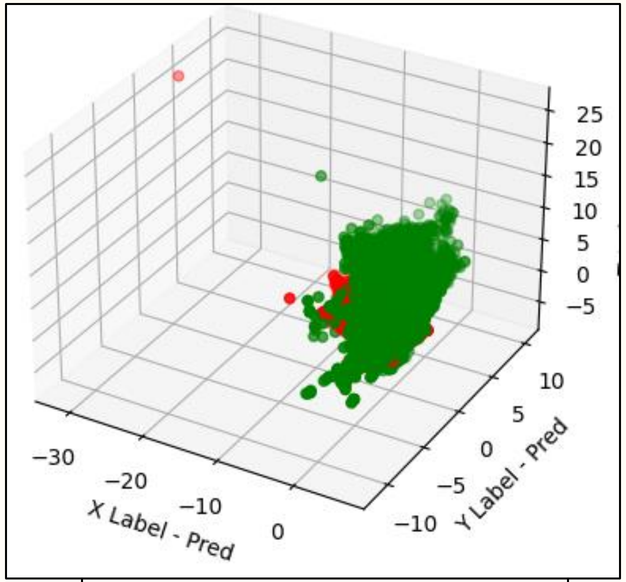
Initial Results

- Mean Squared Logarithmic Error (3 Neurons)
- Accuracy of Autoencoder model: 92.79%

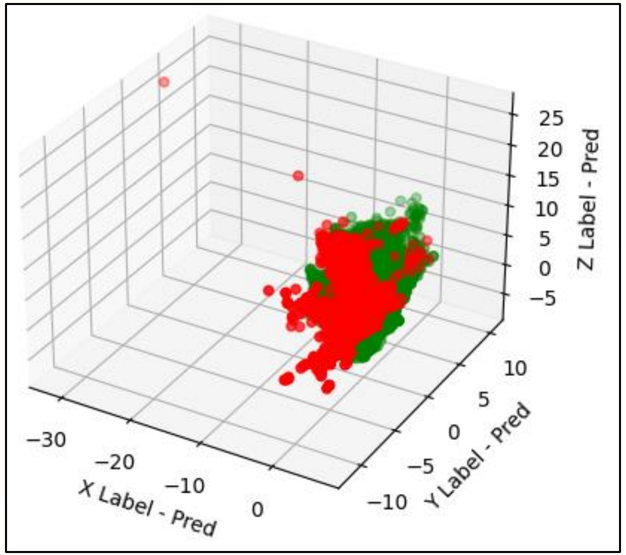


Loss/Val.Loss VS Epochs

N=131724	Pred DoH	Pred HTTPS
Actual DoH	180	4907
Actual HTTPS	20100	106537



Clustering – Ground Truth



Clustering – KMeans

Final Results

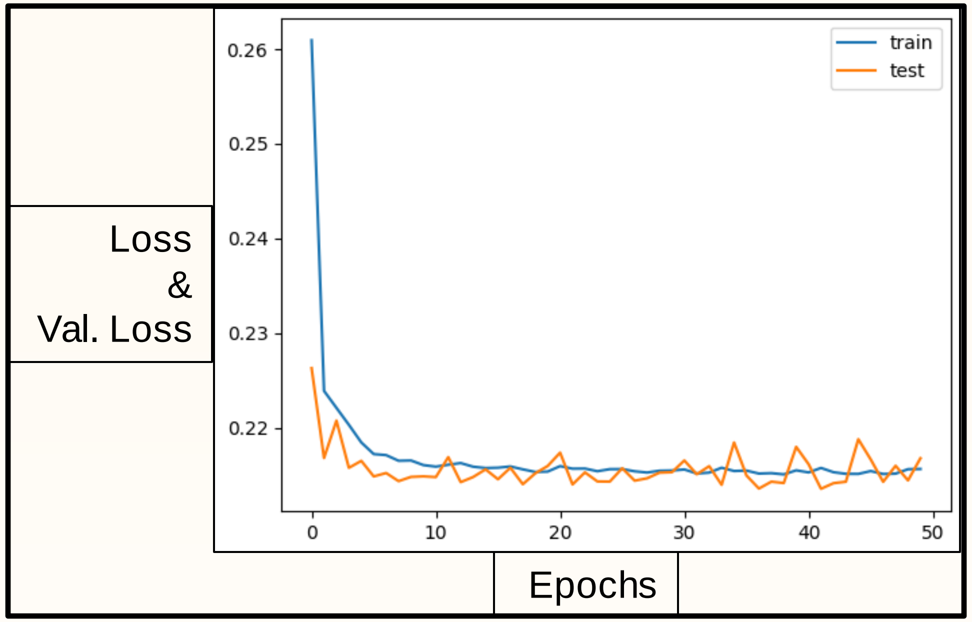
- Binary cross entropy loss function with 3 neurons.
- 50 epochs

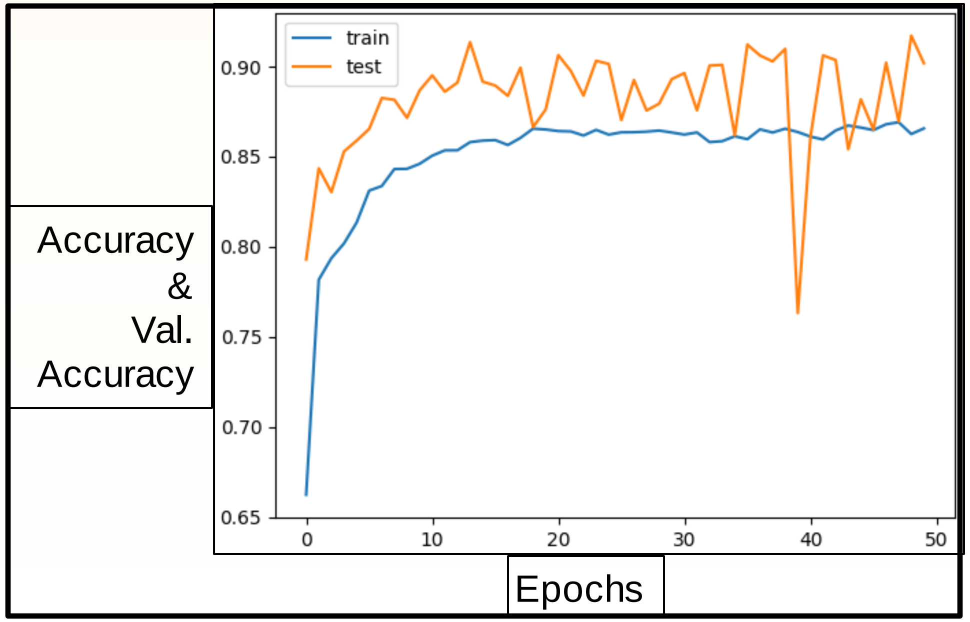
- Accuracy: 98.93%

Confusion matrix

N=131724	Predicted=DoH	Predicted=HTTPS
Actual=DoH	4535	863
Actual=HTTPS	552	125774

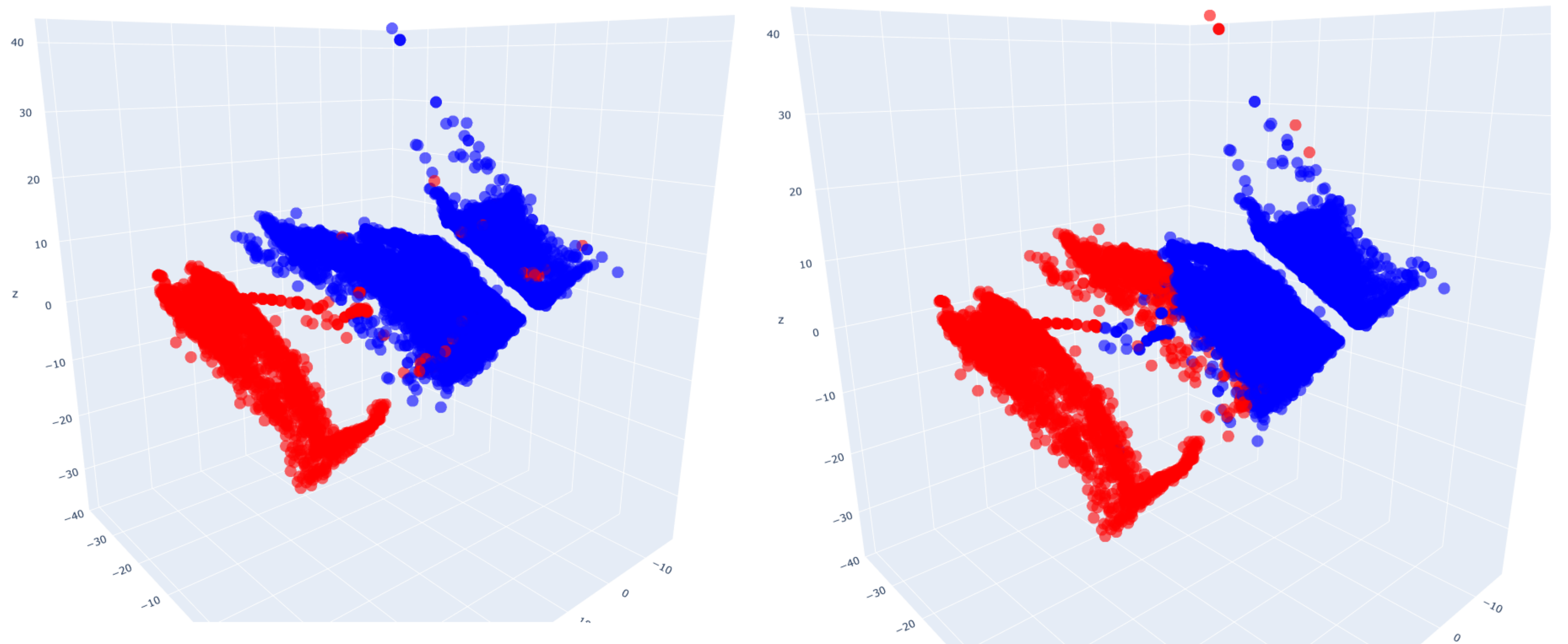
PRECISION: 0.84	RECALL: 0.89	FSCORE: 0.87





Final Results

- Clustering results in 3D



Encoder output plotted in 3D space
Red is DoH
Blue is HTTPS

Labels are based on the Ground Truth

Labels are based on the labels generated by KMeans

Future works

- Collect and create data for botnets using DoH
- Apply the Autoencoder to the collected data

Conclusions

- Tracking botnets using DGA on DoH traffic
- Since data doesn’t exist it’s a difficult task

Questions?

Thank you!

WICHITA STATE UNIVERSITY