

# Data Science & Cybersecurity

Dr. Thomas P. Scanlon

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

# Document Markings

Copyright 2023 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

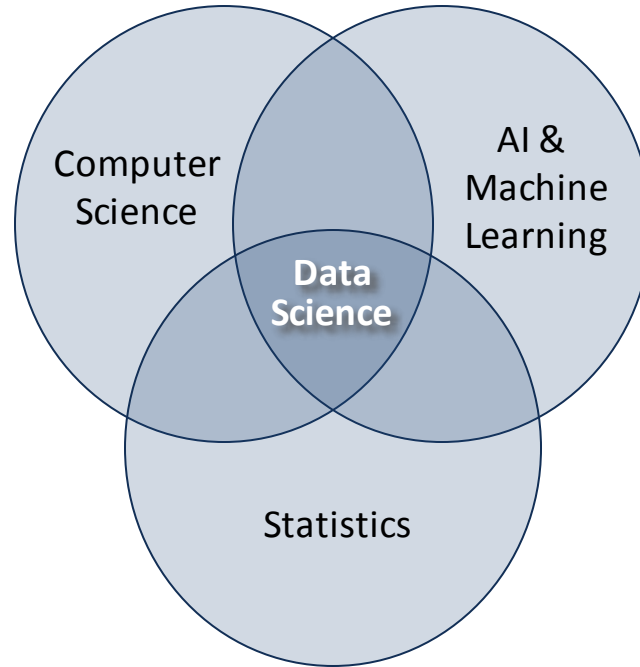
This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

Carnegie Mellon<sup>®</sup> and CERT<sup>®</sup> are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.  
DM23-0052

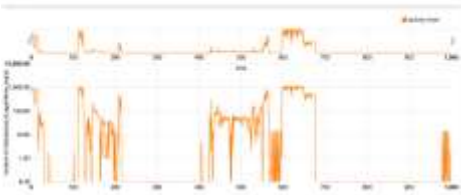
# Application of Data Science to Cybersecurity

*Areas we have worked in*

# Data Science

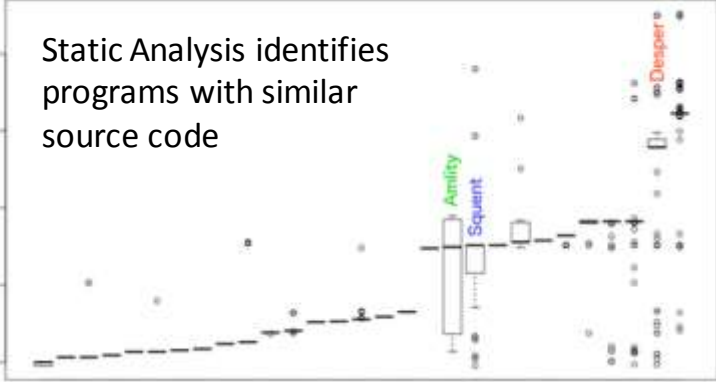
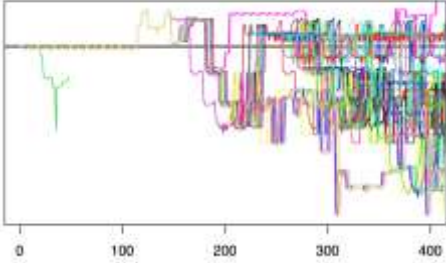


# Malware family classification



Signal Flow graph highlights behavior relating different malware families

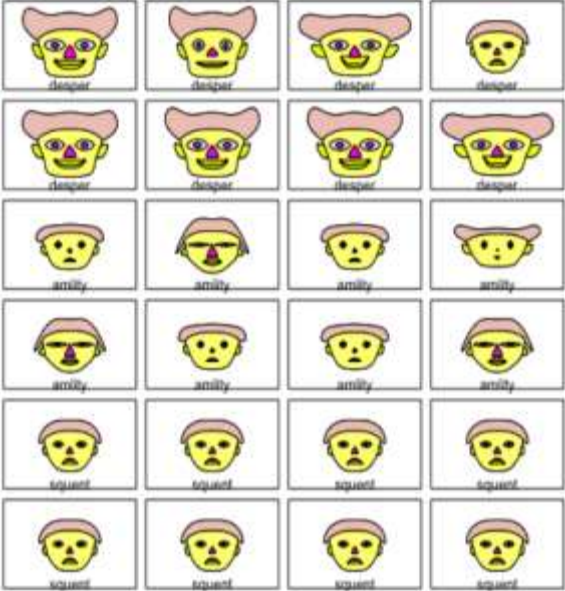
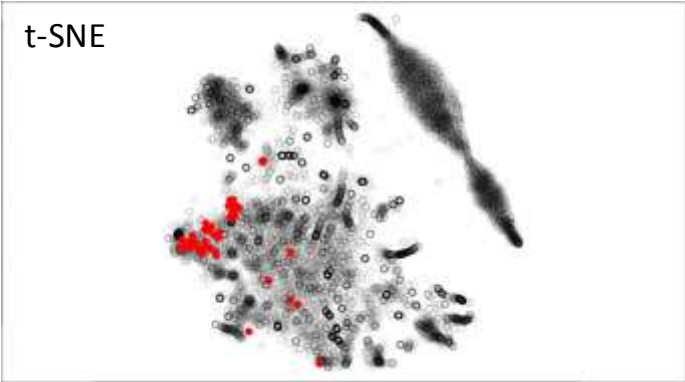
Program instruction analysis shows similarity and diversion of behavior



Static Analysis identifies programs with similar source code

# Malware family classification – Visualization

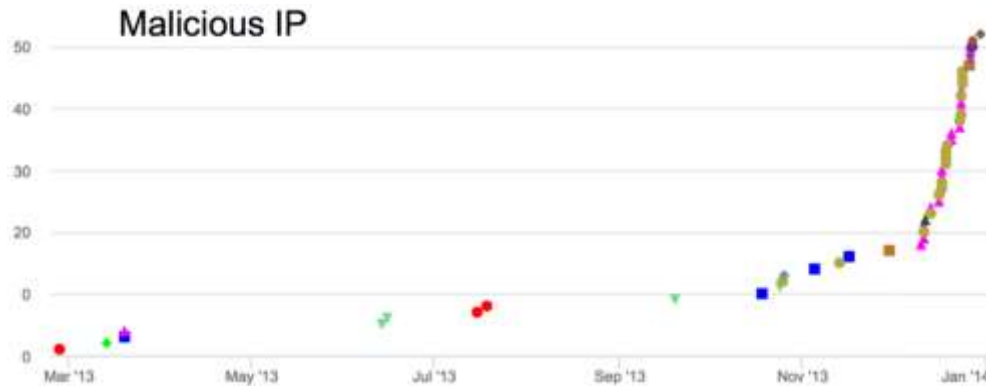
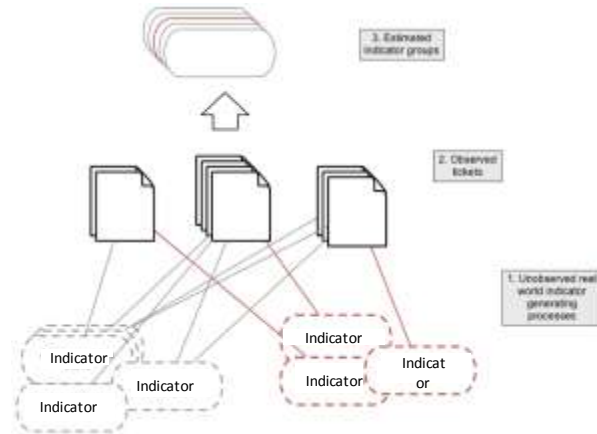
Simplify visualization of extremely complex data through the use of dimensionality reduction and associated visualization techniques



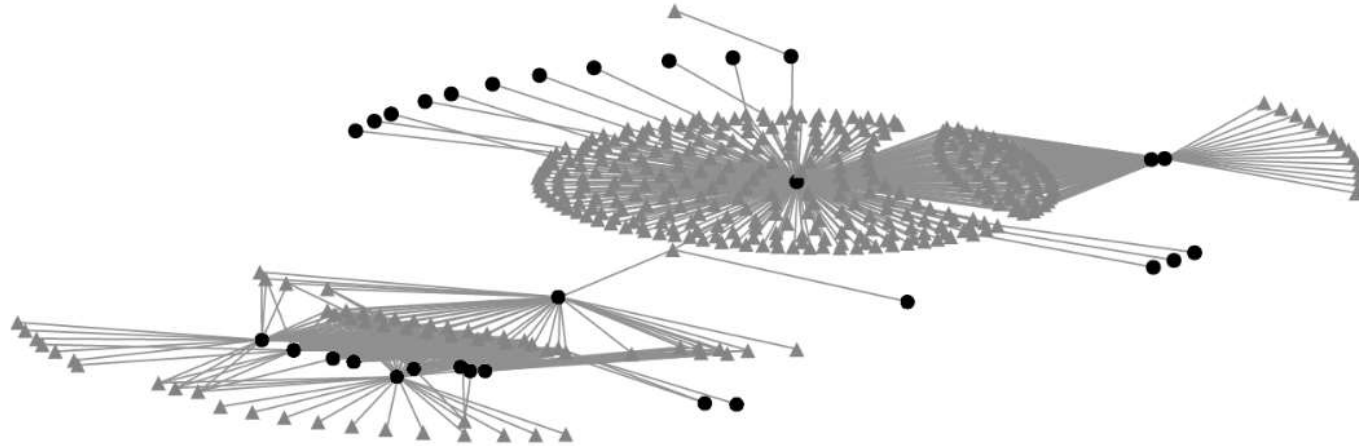
Chernoff face experiment

# Incident Ticket analysis

Collections of incident tickets often contain hidden trends, revealing attacker methods and techniques that are invisible in individual tickets. Natural Language Processing helps find these trends.



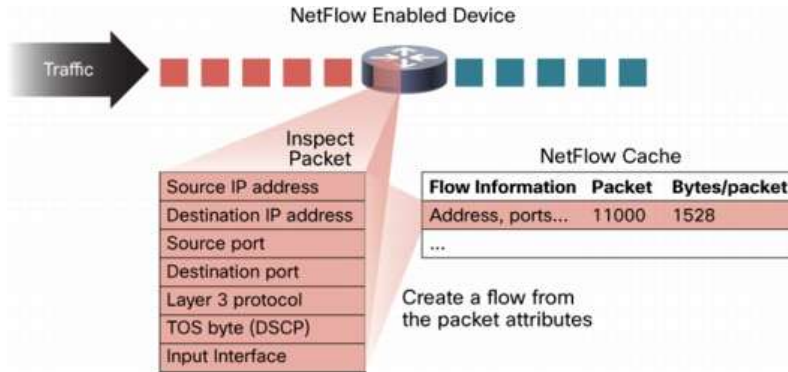
# Incident Ticket analysis – Visualization



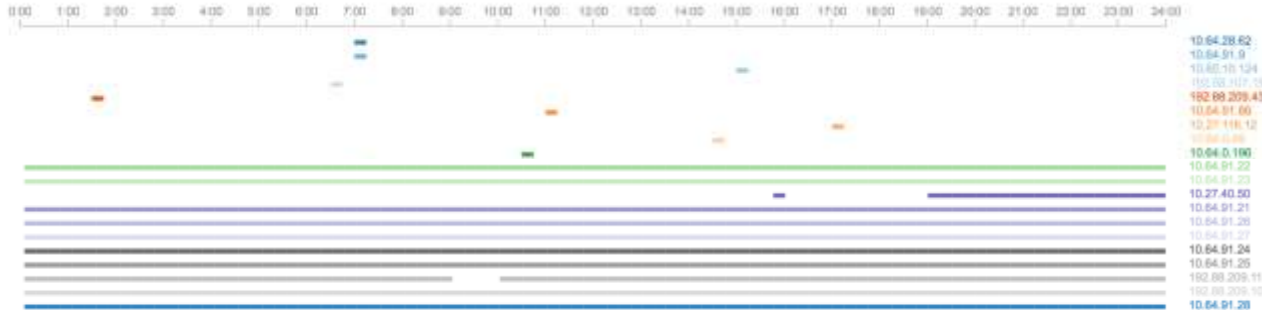
A subset of the ticket-indicator graph  
*(for a small set of selected indicators)*

- Tickets are grey triangles
- Indicators are black circles
- Edges connect tickets to the indicators they contain

# Network traffic classification



Netflow data summarizes network traffic, losing significant information. Numerous techniques exist to infer information from the flows themselves.



# Other Applications



APT Defense

Relation of kinetic and cyber actions

Automated forecasting and detection of cyber-attacks

Static code analyzer behavior

OSINT Analysis

Technical debt estimation

Cognitive support for assurance using Watson

Optimizing planning budgets for travel

Email sentiment analysis

IoT based search-and-rescue

# Application of Data Science to Cybersecurity

*Selected current projects*

# Deepfake LINE

What is a deepfake?

“Believable media generated by a deep neural network”

-- Mirsky and Lee (2020)



**Conceptual Example of a Faceswap Deepfake**

The target's face is placed on the source's face.

# Deepfakes are no longer theoretical threats only

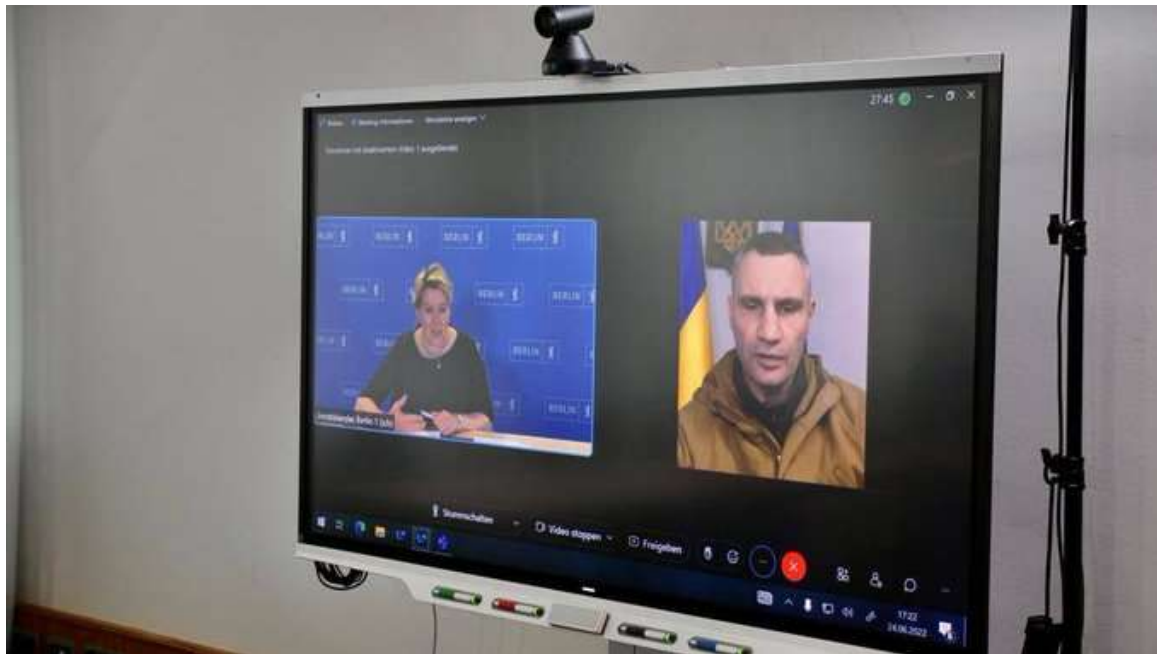


Image from DW.com. The righthand side image is an example of a deepfake used to impersonate the mayor of Kyiv. Brackets are ours.

## Potential Dangers:

- Impersonation of political figures and celebrities
- Defamation of citizens
- Mis-, dis-, and mal-information
- >700k hours of video uploaded to web every day!

**We need robust detectors!**

# And more deepfakes in the news...

## A Deepfake Phone Call Dupes An Employee Into Giving Away \$35 Million

Think your business is too small to be fooled? Think again.

By [Gene Marks](#)

January

## A Telegram bot network is being used to create deepfake nudes

It has produced naked images of at least 104,000 women.

## Deepfakes come to remote job interviews

Is that a real person you're interviewing, or are you talking to a stolen identity?



By [Mike Elgan](#)

Contributing Columnist, Computerworld | JUL 7, 2022 4:30 AM PDT

# But there are differences between real and deepfake images

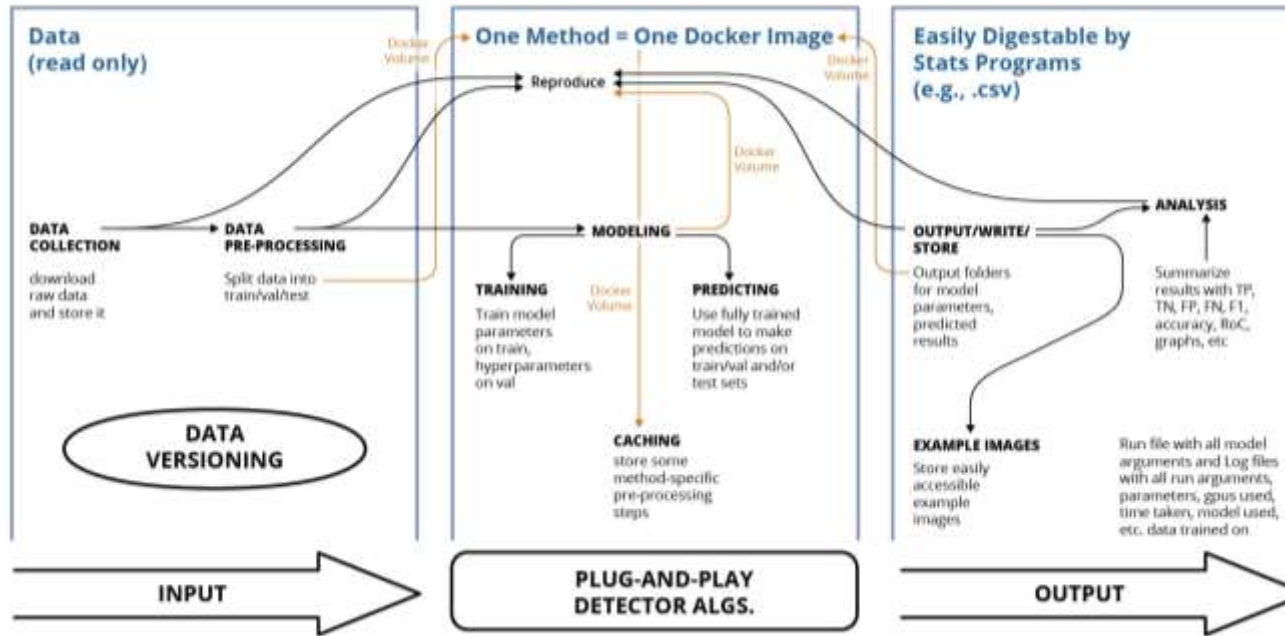
- Hairline
- Edges of eyes
- Corners of mouth
- Chin
- Eyebrows
- Nose

AVG-REAL - AVG-FAKE



# Our AI/ML solution: Deepfake Detection Pipeline (DDP)

## End-to-End Process



DDP is reproducible, portable, and modular

# Future Directions

Current work is focused on deepfake people in images & video

Future directions of work may include:

- Deepfake audio
- Deepfakes objects
- Object detection
- Object classification
- Cyber-physical sensing

# NSA 5G (& Future G) Trust Modeling/Trust Decay

## AI/ML Model Description

- Implement a trust score ( $T$ ) between 0 and 1 for each device
  - 1 = high trust
  - 0 = low trust
- Trust score is a function of
  - Time (probably decreases over time, but not always)
  - Individual device profile
  - Slice profile (or data analytics)

Trust Model Data = Message + UE (device) Profile + Slice Profile  
Security: High

## Message

```
{
  "message": {
    "contents": "Hello World",
    "sender": "10.33.48.19",
    "receiver": "10.29.22.13",
    "time": "2022March21_1420_GMT"
  }
}
```

## UE Profile

```
{
  "trust_data": {
    "capability": "sensor",
    "power": 87,
    "mobile": true,
    "connection": "DSL",
    "hash": "HMAC",
    "device_size": "small",
    "encryption_cap": "AES"
  }
}
```

## Slice Profile

```
{
  "slice_profile": {
    "id": "qiou783",
    "security": "high",
    "n_registered_dev": 242,
    "n_active_dev": 98,
    "n_sent_by": {
      "10.33.48.19": 14
    },
    "trust_scores": [
      {"t0": {"10.33.48.19": 0.72}},
      {"t1": {"10.33.48.19": 0.87}},
      {"tn": {"10.33.48.19": 0.66}}
    ]
  }
}
```

# Trust Score Model

$$T_{SR}(t) = \text{logit}^{-1}(-\gamma^2 t - (\mathbf{X}\boldsymbol{\beta}));$$

If  $T_{SR}(t) < T^*$  then we do not trust sender

such that  $Probability(\text{we trust sender when we shouldn't}) \leq \alpha$

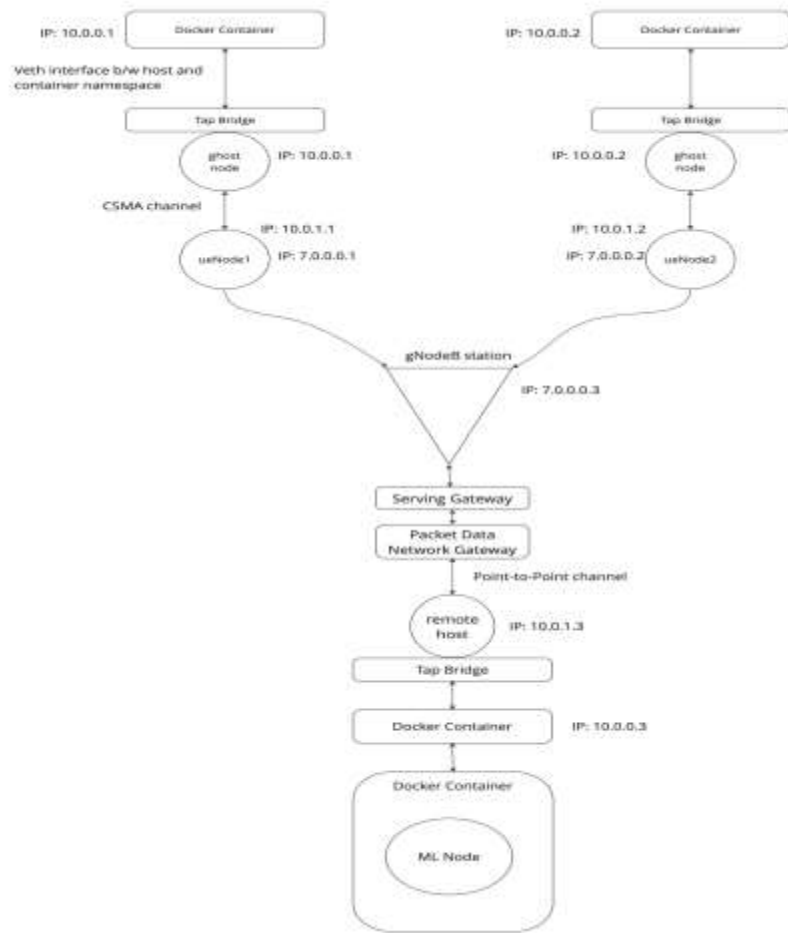
- $T_{SR}(t)$  = trust of sender by receiver at time t
- $T^*$  = minimum trust threshold
- S = sender
- R = receiver
- $\alpha$  in  $[0,1]$  = maximum level of acceptable false positives
- t = time
- $\mathbf{X}$  = other features in metadata (message + UE + slice profile)
- T trust in  $[0,1]$  with 1 = max trust, 0 = no trust
- $(\gamma)$ ,  $(\boldsymbol{\beta})$ ,  $(T^*)$  = parameters to learn

# Simulation Architecture: Overview

A network of nodes are connected in a client-server arrangement.

Packets flow into a server node that collects the data which is fed to an ML model.

Data is outputted as data packets and an ML model trained on the kind of data it received.



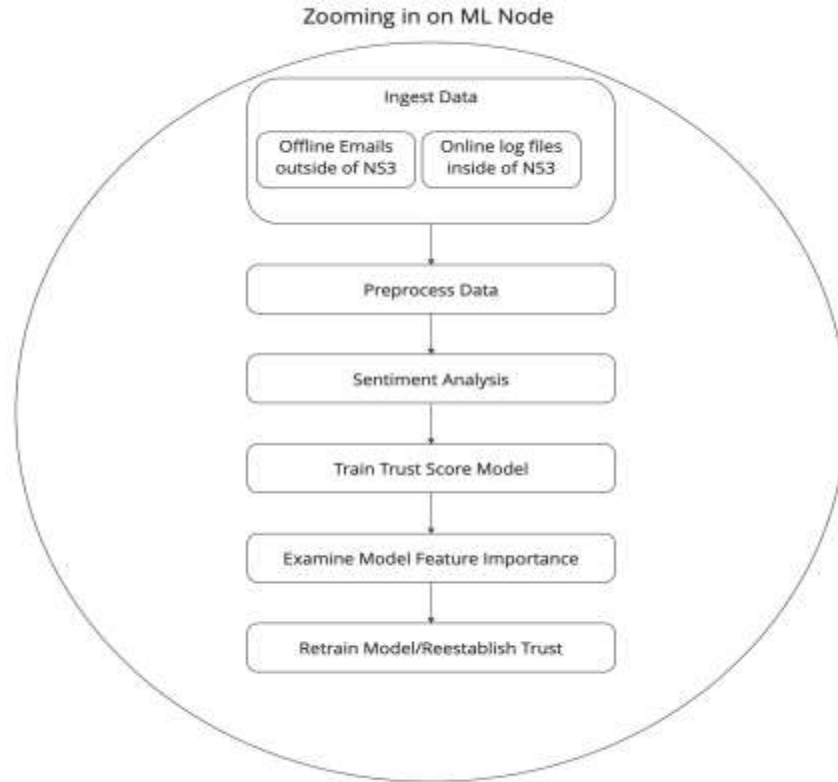
# ML Node

A singular independent and adaptive AI/ML node will act as a quasi-trust manager and track device trustworthiness (trust score) and node behavior.

Organizing that node involves simulating the communication/info exchange between different nodes. We must consider the frequency of info exchange and how to go about asking for info. There will be other trust related parameters as well.

As seen on the previous slide, ML node/trust calculating engine will be in its own (Docker) container. If any nodes are questionable, then we want those nodes to become unavailable in the network, rather than just starting and stopping the data being streamed.

# ML Pipeline



23/10

# Future Directions

Current work is focused on trust of devices on wireless networks

Many areas where trust estimation is relevant:

- Messaging & communication
- OSINT credibility
- Intelligence analysis
- Open-source code trust-ability