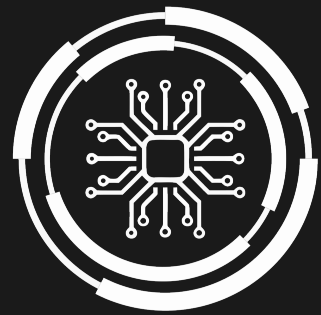


PADME

Manual & Security Audit for Train

ISIC2019II

Karl Kindermann



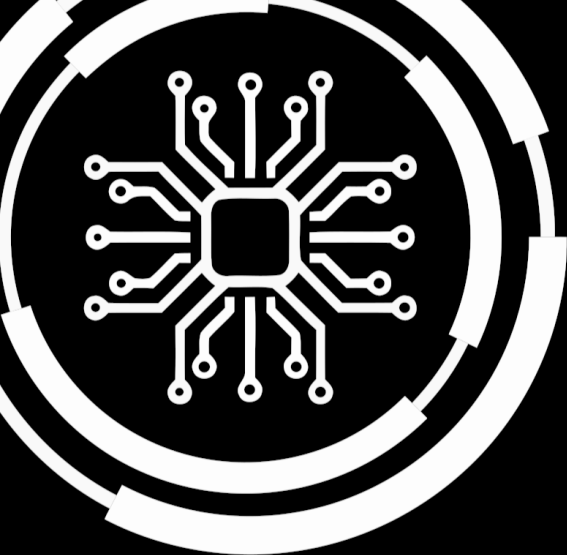
PADME

Copyright © 2023 PADME

PADME-ANALYTICS.DE

Licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License (the “License”). You may not use this file except in compliance with the License. You may obtain a copy of the License at <http://creativecommons.org/licenses/by-nc/3.0>. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

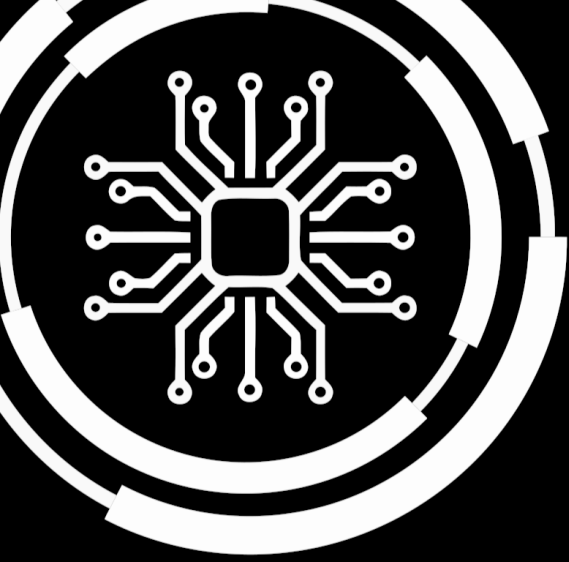
First printing, January 31, 2024



Contents

1	Train Summary	5
1.1	Overview	5
1.2	Image Building File	5
1.3	Requirements & Dependencies	5
1.4	Code Base	6
1.5	Connection Parameters	12
2	Static Train Analysis	13
2.1	SAST	13
2.2	Secret Detection	13
2.3	Dependency Analysis	14
2.4	Blacklist Detection	14
3	Train Image Analysis	15
3.1	Vulnerabilities	15
4	Dynamic Train Analysis	21
4.1	Image size differences after simulation:	21
4.1.1	Simulated route:	21
4.1.2	Directory Overview of changed files:	21
4.1.3	./simulationResults/__config__.cpython-310.pyc	22
4.1.4	./simulationResults/isisdataset.cpython-310.pyc	22
4.1.5	./simulationResults/isisnet.cpython-310.pyc	22
4.1.6	./simulationResults/best_model.pth.tar	22
4.1.7	./simulationResults/checkpoint.pth.tar	22

4.1.8	./simulationResults/val_log.csv	22
4.1.9	./simulationResults/resnet18-f37072fd.pth	22
4.2	Execution metrics:	22
4.3	Network I/O:	22
5	Conclusion	23



1. Train Summary

1.1 Overview

- **Train name:** ISIC2019II
- **Creator of the train:** Karl Kindermann
- **Location:** <https://git.rwth-aachen.de/padme-development/padme-train-depot/-/archive/main/padme-train-depot-main.zip?path=ISIC2019II>

1.2 Image Building File

Listing 1.1: Image building file

```
FROM pytorch/pytorch:2.0.1-cuda11.7-cudnn8-runtime
RUN apt-get update && apt-get install -y git
WORKDIR /usr/src/app

LABEL "envs"="[{"name":"FHIR_SERVER","type":"string",
  "required":true},{"name":"FHIR_PORT","type":"number",
  "required":true},{"name":"BATCH_SIZE","type":"number",
  "required":true},{"name":"LR","type":"string",
  "required":true},{"name":"WEIGHT_DECAY","type":"number",
  "required":true}]"

COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

COPY . .

CMD [ "python", "./main.py" ]
```

1.3 Requirements & Dependencies

Listing 1.2: List of all Requirements

```
requests
Pillow
pandas
numpy
git+https://github.com/beda-software/fhir-py.git
torchvision
datetime
pytz
tqdm
pytest-shutil
```

1.4 Code Base

Legend: env query retrieve_previous_results execute_analysis save log

```
1 from fhirpy import SyncFHIRClient
2 import requests
3 from io import BytesIO
4 import numpy as np
5 import pandas as pd
6 from PIL import Image
7 from torch.utils.data import Dataset
8 import torchvision.transforms as transforms
9
10 class ISICDataset(Dataset):
11     classes = {'NV': 0, 'MEL': 1, 'BKL': 2, 'DF': 3, 'SCC': 4, 'BCC': 5, '
12             VASC': 6, 'AK': 7}
13
14     """ISIC dataset."""
15     def __init__(self, fhir_server, fhir_port, split='train', input_size
16             =256):
17         """
18         Args:
19             fhir_server (string): Address of FHIR Server.
20             fhir_port (string): Port of FHIR Server.
21         """
22         self.fhir_server = fhir_server
23         self.fhir_port = fhir_port
24         self.split = split
25         self.input_size = input_size
26         # Create an instance
27         client = SyncFHIRClient('http://{}:{}/fhir'.format(fhir_server,
28             fhir_port))
29         # Search for patients
30         patients = client.resources('Patient') # Return lazy search set
31         patients_data = []
32         for patient in patients:
33             patient_birthDate = None
34             try:
35                 patient_birthDate = patient.birthDate
36             except:
37                 pass
38             # patinet_id, gender, birthDate
39             patients_data.append([patient.id, patient.gender,
40                 patient_birthDate])
41         patients_df = pd.DataFrame(patients_data, columns=["patient_id", "
42             gender", "birthDate"])
43         # Search for media
```

```

39     media_list = client.resources('Media').include('Patient', 'subject'
40 )
41     media_data = []
42     for media in media_list:
43         media_bodySite = None
44         media_reasonCode = None
45         media_note = None
46         try:
47             media_bodySite = media.bodySite.text
48         except:
49             pass
50         try:
51             media_reasonCode = media.reasonCode[0].text
52         except:
53             pass
54         try:
55             media_note = media.note[0].text
56         except:
57             pass
58         media_data.append([media.subject.id, media.id, media_bodySite,
59 media_reasonCode, media_note, media.content.url])
60     media_df = pd.DataFrame(media_data, columns=["patient_id", "
61 media_id", "bodySite", "reasonCode", "note", "image_url"])
62     self.data_df = pd.merge(patients_df, media_df, on='patient_id', how
63 ='outer')
64     self.data_df = self.data_df[self.data_df['note'].notna()].
65     reset_index()
66     self.trans = transforms.Compose([transforms.RandomHorizontalFlip(),
67                                     transforms.RandomVerticalFlip(),
68                                     transforms.ColorJitter(brightness
69 =32. / 255., saturation=0.5),
70                                     transforms.Resize(self.input_size)
71                                     ,
72                                     transforms.ToTensor()])
73
74     def __len__(self):
75         if self.split == "train":
76             return int(len(self.data_df) * 0.8)
77         elif self.split == "val":
78             return int(len(self.data_df) * 0.2)
79
80     def __getitem__(self, idx):
81         val_start_id = int(len(self.data_df) * 0.8)
82         if self.split == "train":
83             idx = idx
84         else:
85             idx = idx + val_start_id
86         img_url = self.data_df.loc[idx, 'image_url']
87         note = self.data_df.loc[idx, 'note']
88         y = self.classes[note]
89         img_res = requests.get(img_url)
90         if img_res.status_code == 200:
91             x = Image.open(BytesIO(img_res.content))
92             x = self.center_crop(x)
93             x = self.trans(x)
94             # Transform y
95             y = np.int64(y)
96             return {"image": x, "label": y}
97         else:
98             raise RuntimeError("Image url {} is not reachable!".format(
99 img_url))

```

```

92
93     def center_crop(self, pil_img):
94         img_width, img_height = pil_img.size
95         if img_width > img_height:
96             crop_size = img_height
97         else:
98             crop_size = img_width
99         return pil_img.crop(((img_width - crop_size) // 2,
100                             (img_height - crop_size) // 2,
101                             (img_width + crop_size) // 2,
102                             (img_height + crop_size) // 2))

```

Listing 1.3: isicdataset.py

```

1 import os
2 import os.path as osp
3 import numpy as np
4
5 import glob
6 import tqdm
7 import shutil
8 import pytz
9 import datetime
10
11 import torch
12 import torch.nn as nn
13 from torch.autograd import Variable
14
15 from isicnet import ISICNet
16 from isicdataset import ISICDataset
17
18 ## Define evaluation function
19 def _fast_hist(label_true, label_pred, n_class):
20     hist = np.bincount(
21         n_class * label_true.astype(int) +
22         label_pred.astype(int), minlength=n_class ** 2).reshape(n_class,
23         n_class)
24     return hist
25
26 def label_accuracy_score(label_trues, label_preds, n_class=8):
27     hist = np.zeros((n_class, n_class))
28     hist += _fast_hist(label_trues, label_preds, n_class)
29     acc = np.diag(hist).sum() / hist.sum()
30     with np.errstate(divide='ignore', invalid='ignore'):
31         precision = np.diag(hist) / hist.sum(axis=1)
32     mean_precision = np.nanmean(precision)
33     with np.errstate(divide='ignore', invalid='ignore'):
34         recall = np.diag(hist) / hist.sum(axis=0)
35     mean_recall = np.nanmean(recall)
36     with np.errstate(divide='ignore', invalid='ignore'):
37         iou = np.diag(hist) / (hist.sum(axis=1) + hist.sum(axis=0) - np.
38         diag(hist))
39     mean_iou = np.nanmean(iou)
40     with np.errstate(divide='ignore', invalid='ignore'):
41         f1 = (2 * np.diag(hist)) / (hist.sum(axis=1) + hist.sum(axis=0) + 2
42         * np.diag(hist))
43     mean_f1 = np.nanmean(f1)
44     return acc, mean_precision, mean_recall, mean_iou, mean_f1
45
46 ## Define directory of output
47 here = osp.dirname(osp.abspath(__file__))

```



```

45 out_dir = osp.join(here, 'output')
46 if not os.path.exists(out_dir):
47     os.makedirs(out_dir)
48 runs = sorted(glob.glob(os.path.join(out_dir, 'run_*')))
49 run_id = int(runs[-1].split('_')[-1]) + 1 if runs else 0
50 experiment_dir = os.path.join(out_dir, 'run_{}'.format(str(run_id)))
51 if not os.path.exists(experiment_dir):
52     os.makedirs(experiment_dir)
53
54 ## Define (input) variables from Docker Container environment variables
55 fhir_server = str(os.environ['FHIR_HOST_NAME'])
56 fhir_port = str(os.environ['FHIR_PORT'])
57 # num_station = int(os.environ['NUM_STATION'])
58 # sid = int(os.environ['SID'])
59 #Hyperparameters
60 batch_size = int(os.environ.get('BATCH_SIZE', '1'))
61 num_epoch = int(os.environ.get('NUM_EPOCH', '1'))
62 lr = float(os.environ.get('LR', '0.01'))
63 weight_decay = float(os.environ.get('WEIGHT_DECAY', '0.005'))
64 model_name = "resnet18"
65
66 ## Define (output) file formats
67 if not osp.exists(osp.join(experiment_dir, 'val_log.csv')):
68     with open(osp.join(experiment_dir, 'val_log.csv'), 'w') as f:
69         header = ['epoch', 'Loss', 'Acc', 'Precision', 'Recall', 'Iou', '
        F1Score', 'train/Loss', 'elapsed_time']
70         header = map(str, header)
71         f.write(','.join(header) + '\n')
72         print("Initial Log file")
73
74 cuda = torch.cuda.is_available()
75 torch.manual_seed(1337)
76 if cuda:
77     torch.cuda.manual_seed(1337)
78
79 ## Initial Model
80 print("Initial Model")
81 model = ISICNet(backbone=model_name)
82 print("Initial Model {}".format(model_name))
83 if cuda:
84     print("Cuda:", cuda)
85     model = model.cuda()
86 else:
87     print("Running on CPU")
88 ## Initial Datasets of train and val on station 1, 2, 3 and test
89 kwargs = {'num_workers': 4, 'pin_memory': True} if cuda else {}
90 print("Initial Training Dataset")
91 train_dataloader = torch.utils.data.DataLoader(ISICDataset(fhir_server,
        fhir_port, split='train'), batch_size=batch_size, shuffle=True, **
        kwargs)
92 print("Initial Val Dataset")
93 val_dataloader = torch.utils.data.DataLoader(ISICDataset(fhir_server,
        fhir_port, split='val'), batch_size=batch_size, shuffle=False, **kwargs
        )
94 ## Initial criterion (Cross Entropy Loss)
95 print("Initial Loss function")
96 criterion = nn.CrossEntropyLoss()
97 ## Initial Optimizers for station
98 print("Initial Optimizer")
99 optim = torch.optim.Adam(model.parameters(), lr=lr, weight_decay=
        weight_decay)

```

```

100
101 ## Load model from previous train
102 if run_id > 0:
103     prev_experiment_dir = osp.join(out_dir, 'run_{}'.format(str(run_id - 1)
104     ))
105     if osp.exists(osp.join(prev_experiment_dir, 'best_model.pth.tar')):
106         prev_best_model = torch.load(osp.join(prev_experiment_dir, '
107         best_model.pth.tar'))
108         model.load_state_dict(prev_best_model['model_state_dict'])
109         optim.load_state_dict(prev_best_model['optim_state_dict'])
110         shutil.copy(osp.join(prev_experiment_dir, 'best_model.pth.tar'),
111                     osp.join(experiment_dir, 'best_model.pth.tar'))
112         print("Model loaded from previous train.")
113     else:
114         print("No previous best model found!")
115 else:
116     torch.save({
117         'epoch': 0,
118         'optim_state_dict': optim.state_dict(),
119         'model_state_dict': model.state_dict(),
120         'best_acc': 0.0,
121     }, osp.join(experiment_dir, 'best_model.pth.tar'))
122
123 timestamp_start = datetime.datetime.now(pytz.timezone('Asia/Tokyo'))
124 best_acc = 0.0
125 ## Run the training processing on the station
126 for epoch in range(num_epoch):
127     model.train()
128     train_loss = 0.0
129     for batch_idx, sample in tqdm.tqdm(enumerate(train_dataloader), total=
130     len(train_dataloader), desc='Station Train epoch=%d' % epoch, ncols=80,
131     leave=False):
132         assert model.training
133         img, lbl = sample['image'], sample['label']
134         if cuda:
135             img, lbl = img.cuda(), lbl.cuda()
136             img, lbl = Variable(img), Variable(lbl)
137             optim.zero_grad()
138             pred = model(img)
139             loss = criterion(pred, lbl)
140             train_loss = train_loss + loss.data.item()
141             loss.backward()
142             optim.step()
143
144     train_loss = train_loss / len(train_dataloader)
145     print("Train epoch {} finished with average train loss of {}".format(
146     epoch, train_loss))
147
148     model.eval()
149     val_loss = 0.0
150     label_trues, label_preds = [], []
151     for batch_idx, sample in tqdm.tqdm(enumerate(val_dataloader), total=len
152     (val_dataloader), desc='Station Val epoch=%d' % epoch, ncols=80, leave=
153     False):
154         img, lbl = sample['image'], sample['label']
155         if cuda:
156             img, lbl = img.cuda(), lbl.cuda()
157             img, lbl = Variable(img), Variable(lbl)
158             with torch.no_grad():
159                 pred = model(img)
160                 loss = criterion(pred, lbl)

```

```

154     val_loss = val_loss + loss.data.item()
155     lbl = lbl.data.cpu().numpy()
156     pred = pred.data.max(1)[1].cpu().numpy()
157     label_trues = np.concatenate((label_trues, lbl), axis=0)
158     label_preds = np.concatenate((label_preds, pred), axis=0)
159     val_loss = val_loss / len(val_dataloader)
160     acc, mean_precision, mean_recall, mean_iou, mean_f1 =
label_accuracy_score(label_trues, label_preds)
161     with open(osp.join(experiment_dir, 'val_log.csv'), 'a') as f:
162         elapsed_time = (datetime.datetime.now(pytz.timezone('Asia/Tokyo'))
- timestamp_start).total_seconds()
163         log = [epoch, val_loss, acc, mean_precision, mean_recall, mean_iou,
mean_f1, train_loss, elapsed_time]
164         log = map(str, log)
165         f.write(','.join(log) + '\n')
166
167     is_best = acc > best_acc
168     if is_best:
169         best_acc = acc
170     torch.save({
171         'epoch': epoch,
172         'optim_state_dict': optim.state_dict(),
173         'model_state_dict': model.state_dict(),
174         'best_acc': best_acc,
175     }, osp.join(experiment_dir, 'checkpoint.pth.tar'))
176     if is_best:
177         shutil.copy(osp.join(experiment_dir, 'checkpoint.pth.tar'), osp.
join(experiment_dir, 'best_model.pth.tar'))
178     print("Station Val epoch {} finished with loss of {}, acc of {},
precision of {}, recall of {}, iou of {}, f1-score of {}".format(epoch
, val_loss, acc, mean_precision, mean_recall, mean_iou, mean_f1))
179 print("Finished training process")

```

Listing 1.4: main.py

```

1 import torch
2 import torch.nn as nn
3 from torchvision.models import resnet
4
5 class ISICNet(nn.Module):
6     def __init__(self, n_feature=3, n_class=8, backbone='resnet18'):
7         super(ISICNet, self).__init__()
8         self.n_feature = n_feature
9         self.n_class = n_class
10        self.backbone = backbone
11        if self.backbone == 'resnet18':
12            base = resnet.resnet18(pretrained=True)
13            self.resnet_expansion = 1
14            self.in_block = nn.Sequential(
15                nn.Conv2d(self.n_feature, 64, kernel_size=(7, 7), stride=(2, 2)
, padding=(3, 3), bias=False),
16                base.bn1,
17                base.relu,
18                base.maxpool)
19            self.encoder1 = base.layer1
20            self.encoder2 = base.layer2
21            self.encoder3 = base.layer3
22            self.encoder4 = base.layer4
23            self.avgpool = base.avgpool
24            self.flatten = nn.Flatten()
25            self.fc = nn.Linear(512*self.resnet_expansion, self.n_class , bias=

```

```

True)
26
27     def forward(self, x):
28         h = self.in_block(x)
29         h = self.encoder1(h)
30         h = self.encoder2(h)
31         h = self.encoder3(h)
32         h = self.encoder4(h)
33         y = self.fc(self.flatten(self.avgpool(h)))
34         return y
35
36 if __name__ == "__main__":
37     model = ISICNet().cuda()
38     x = torch.rand(2, 3, 512, 512).cuda()
39     y = model(x)
40     print(y)
41     print(y.data.max(1)[1])
42     # t = torch.tensor([1.0, 0.0, 1.0])
43     # y = torch.tensor([0.02, 0.05, 0.99])
44     # bce = nn.BCELoss()
45     # l = torch.log(torch.tensor(0.02))+ torch.log(torch.tensor(1-0.05)) +
46     torch.log(torch.tensor(0.99))
47     # print(bce(y, t))
48     # print(l)

```

Listing 1.5: isicnet.py

```

1 {"resources": [{"id": "5bce186a-5005-4ca8-878e-9d8d4e0747c0", "datasets": [{"id":
  ": "0ed80d8f-6f34-45c7-979f-714681f59046a"}, {"id": "a80bac0f-7f44-4fdd-
  a3c5-45f9aad175b6"}]}], "route": [{"id": "5bce186a-5005-4ca8-878e-9
  d8d4e0747c0", "ownEnvs": []}]}

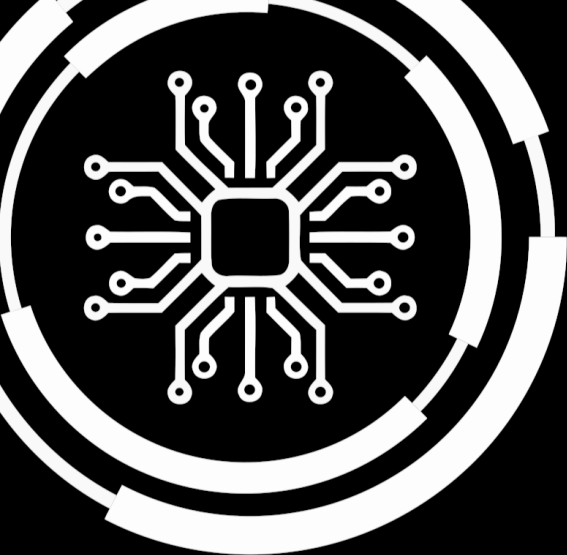
```

Listing 1.6: route.json

1.5 Connection Parameters

Overall, 2 connection parameters have been found in the train:

Name	Type	Required?
FHIR_SERVER	string	True
FHIR_PORT	number	True



2. Static Train Analysis

Summary 2.1

- **Number of Lines:** 223754
- **Vulnerabilities per Line:** 0.0
- **Analysis Score:** 1.0

Total

Summary: **Low: 0** **Medium: 0** **High: 0**

2.1 SAST

Summary: **Low: 0** **Medium: 0** **High: 0**

Detailed Vulnerabilities:

Vulnerability 2.1

Test-specific ID: 62d9a0c9581c461f41278053d31b06a1ee7b663f285f97c18d2b90af109a170c

Severity: Info

Identifiers:

- **semgrep_id:** bandit.B101
- **cwe:** 754
- **bandit_test_id:** B101

Message: Improper Check for Unusual or Exceptional Conditions

File: ISIC2019II/main.py

Start line: 128

2.2 Secret Detection

Summary: **Low: 0** **Medium: 0** **High: 0**

Detailed Secret Detection:

2.3 Dependency Analysis

Summary: Low: 0 Medium: 0 High: 0

Detailed Dependency Analysis:

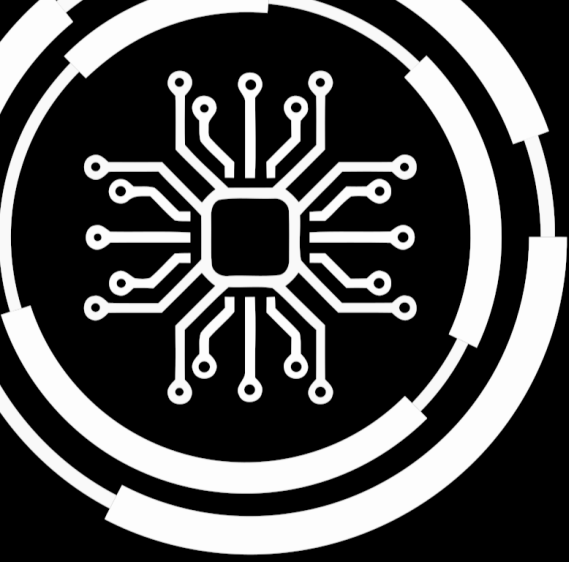
2.4 Blacklist Detection

The following patterns have been configured:

\bPOST\b

The following matches have been found:

None



3. Train Image Analysis

3.1 Vulnerabilities

Summary: Low: 22 Medium: 8 High: 0 Critical: 0

Detailed Vulnerabilities:

Critical:

High:

Medium:

Vulnerability 3.1

Test-specific ID: SNYK-UBUNTU2004-GNUTLS28-6069042 Severity: medium

Identifiers: Message:Information Exposure

Vulnerability 3.2

Test-specific ID: SNYK-UBUNTU2004-GNUTLS28-6172192 Severity: medium

Identifiers: Message:Information Exposure

Vulnerability 3.3

Test-specific ID: SNYK-UBUNTU2004-NCURSES-5423141 Severity: medium

Identifiers: Message:Out-of-bounds Write

Vulnerability 3.4

Test-specific ID: SNYK-UBUNTU2004-OPENSSSH-6142083 **Severity:** medium

Identifiers: **Message:**CVE-2023-51767

Vulnerability 3.5

Test-specific ID: SNYK-UBUNTU2004-OPENSSL-5661533 **Severity:** medium

Identifiers: **Message:**Allocation of Resources Without Limits or Throttling

Vulnerability 3.6

Test-specific ID: SNYK-UBUNTU2004-PAM-6170207 **Severity:** medium

Identifiers: **Message:**CVE-2024-22365

Vulnerability 3.7

Test-specific ID: SNYK-UBUNTU2004-TAR-6096088 **Severity:** medium

Identifiers: **Message:**CVE-2023-39804

Vulnerability 3.8

Test-specific ID: SNYK-UBUNTU2004-XZUTILS-5854646 **Severity:** medium

Identifiers: **Message:**CVE-2020-22916

Low:

Vulnerability 3.9

Test-specific ID: SNYK-UBUNTU2004-COREUTILS-583876 **Severity:** low

Identifiers: **Message:**Improper Input Validation

Vulnerability 3.10

Test-specific ID: SNYK-UBUNTU2004-GIT-580979 **Severity:** low

Identifiers: **Message:**Improper Input Validation

Vulnerability 3.11

Test-specific ID: SNYK-UBUNTU2004-GLIBC-1297554 **Severity:** low

Identifiers: **Message:**Use After Free

Vulnerability 3.12

Test-specific ID: SNYK-UBUNTU2004-GLIBC-2415100 **Severity:** low

Identifiers: **Message:**Allocation of Resources Without Limits or Throttling

Vulnerability 3.13

Test-specific ID: SNYK-UBUNTU2004-GLIBC-5901432 **Severity:** low

Identifiers: **Message:**Use After Free

Vulnerability 3.14

Test-specific ID: SNYK-UBUNTU2004-GLIBC-5901476 **Severity:** low

Identifiers: **Message:**Use After Free

Vulnerability 3.15

Test-specific ID: SNYK-UBUNTU2004-GNUPG2-3035407 **Severity:** low

Identifiers: **Message:**Out-of-bounds Write

Vulnerability 3.16

Test-specific ID: SNYK-UBUNTU2004-KRB5-579303 **Severity:** low

Identifiers: **Message:**Integer Overflow or Wraparound

Vulnerability 3.17

Test-specific ID: SNYK-UBUNTU2004-LIBPNG16-3124878 **Severity:** low

Identifiers: **Message:**NULL Pointer Dereference

Vulnerability 3.18

Test-specific ID: SNYK-UBUNTU2004-NCURSES-1656318 **Severity:** low

Identifiers: **Message:**Out-of-bounds Write

Vulnerability 3.19

Test-specific ID: SNYK-UBUNTU2004-NCURSES-2770341 **Severity:** low

Identifiers: **Message:**Out-of-bounds Read

Vulnerability 3.20

Test-specific ID: SNYK-UBUNTU2004-OPENSSH-1047872 **Severity:** low

Identifiers: **Message:**Information Exposure

Vulnerability 3.21

Test-specific ID: SNYK-UBUNTU2004-OPENSSL-5786273 **Severity:** low

Identifiers: **Message:**Inefficient Regular Expression Complexity

Vulnerability 3.22

Test-specific ID: SNYK-UBUNTU2004-OPENSSL-5811821 **Severity:** low

Identifiers: **Message:**Excessive Iteration

Vulnerability 3.23

Test-specific ID: SNYK-UBUNTU2004-PATCH-2325780 **Severity:** low

Identifiers: **Message:**Release of Invalid Pointer or Reference

Vulnerability 3.24

Test-specific ID: SNYK-UBUNTU2004-PATCH-582546 **Severity:** low

Identifiers: **Message:**Double Free

Vulnerability 3.25

Test-specific ID: SNYK-UBUNTU2004-PCRE3-580031 **Severity:** low

Identifiers: **Message:**Uncontrolled Recursion

Vulnerability 3.26

Test-specific ID: SNYK-UBUNTU2004-PROCPS-5816664 **Severity:** low

Identifiers: **Message:**Out-of-bounds Write

Vulnerability 3.27

Test-specific ID: SNYK-UBUNTU2004-SHADOW-5425687 **Severity:** low

Identifiers: **Message:**Arbitrary Code Injection

Vulnerability 3.28

Test-specific ID: SNYK-UBUNTU2004-SHADOW-577863 **Severity:** low

Identifiers: **Message:**Time-of-check Time-of-use (TOCTOU)

Vulnerability 3.29

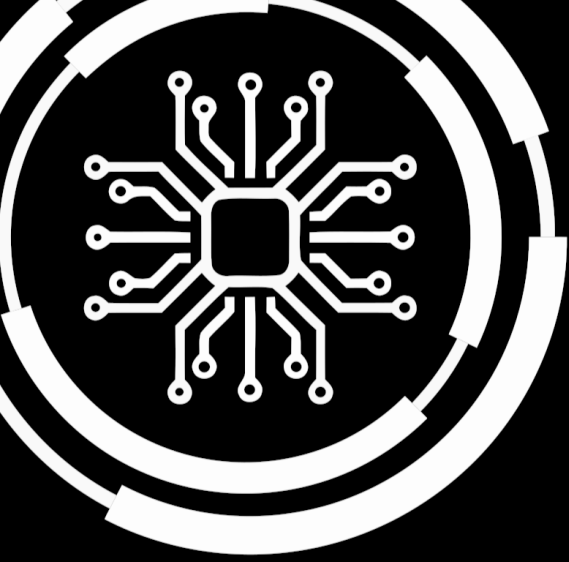
Test-specific ID: SNYK-UBUNTU2004-SYSTEMD-3339226 **Severity:** low

Identifiers: **Message:**CVE-2023-26604

Vulnerability 3.30

Test-specific ID: SNYK-UBUNTU2004-SYSTEMD-6137854 **Severity:** low

Identifiers: **Message:**CVE-2023-7008



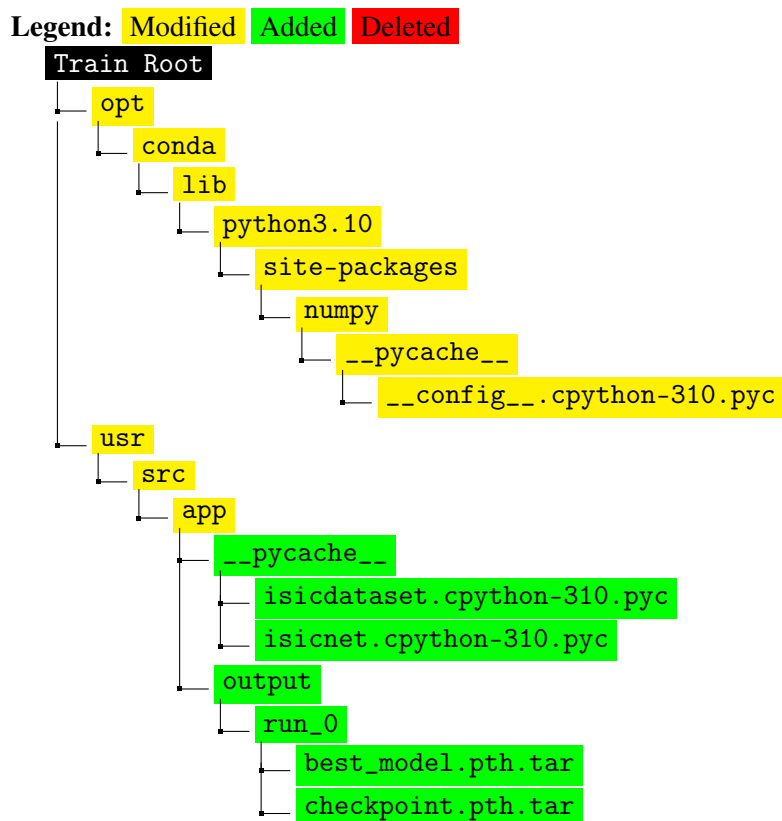
4. Dynamic Train Analysis

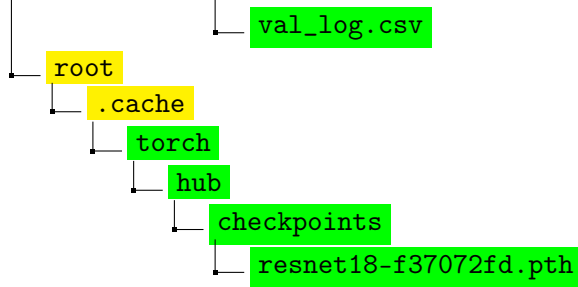
4.1 Image size differences after simulation:

4.1.1 Simulated route:

- Station 5bce186a-5005-4ca8-878e-9d8d4e0747c0: 225937765

4.1.2 Directory Overview of changed files:





4.1.3 ./simulationResults/__config__.cpython-310.pyc

No Preview Available! https://git.rwth-aachen.de/padme-development/padme-train-depot/-/jobs/4570446/artifacts/file/simulationResults/__config__.cpython-310.pyc

4.1.4 ./simulationResults/isicdataset.cpython-310.pyc

No Preview Available! <https://git.rwth-aachen.de/padme-development/padme-train-depot/-/jobs/4570446/artifacts/file/simulationResults/isicdataset.cpython-310.pyc>

4.1.5 ./simulationResults/isicnet.cpython-310.pyc

No Preview Available! <https://git.rwth-aachen.de/padme-development/padme-train-depot/-/jobs/4570446/artifacts/file/simulationResults/isicnet.cpython-310.pyc>

4.1.6 ./simulationResults/best_model.pth.tar

No Preview Available! https://git.rwth-aachen.de/padme-development/padme-train-depot/-/jobs/4570446/artifacts/file/simulationResults/best_model.pth.tar

4.1.7 ./simulationResults/checkpoint.pth.tar

No Preview Available! <https://git.rwth-aachen.de/padme-development/padme-train-depot/-/jobs/4570446/artifacts/file/simulationResults/checkpoint.pth.tar>

4.1.8 ./simulationResults/val_log.csv

Listing 4.1: val_log.csv

```
epoch, Loss, Acc, Precision, Recall, Iou, F1Score, train/Loss,
elapsed_time
0, 714102.0, 0.0, 0.0, 0.0, 0.0, 0.0, 7.464000657200813, 1.320086
```

4.1.9 ./simulationResults/resnet18-f37072fd.pth

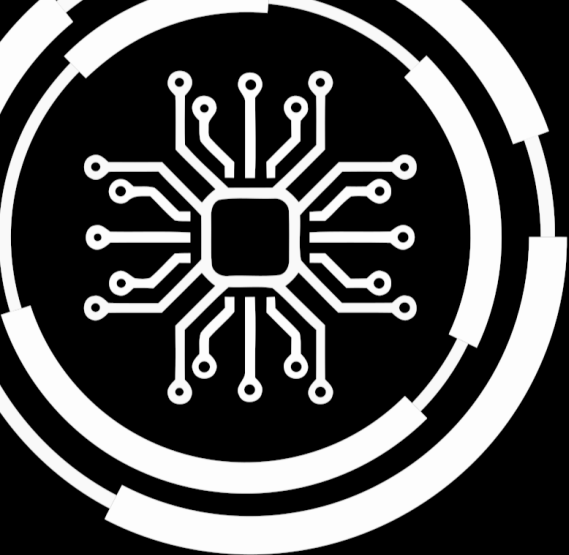
No Preview Available! <https://git.rwth-aachen.de/padme-development/padme-train-depot/-/jobs/4570446/artifacts/file/simulationResults/resnet18-f37072fd.pth>

4.2 Execution metrics:

- Maximum CPU Usage:5.8371202223355425
- Maximum Memory Usage:562102272
- Maximum Number of PIDS:11

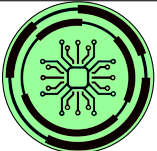

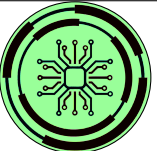
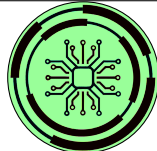
4.3 Network I/O:

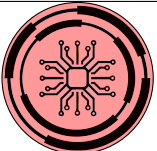
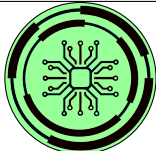
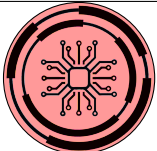
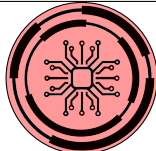
- RX (Reading):47501630
- TX (Transmitting):340878



5. Conclusion

The PADME Security Audit has come to the following conclusion:

Static Complete	SAST	Dependencies	Secret Detection
			

Standard Compliant	Blacklist	Train Image Analysis	DAST
			

Warnings

- Identified network traffic in dynamic analysis.
- Couldn't detect compliance with standards (usage of the python module `padme_conductor` [<https://pypi.org/project/padme-conductor>]).

Audit Result

Train Rejected

January 31, 2024

