digital.security

WARS OF THE MACHINES

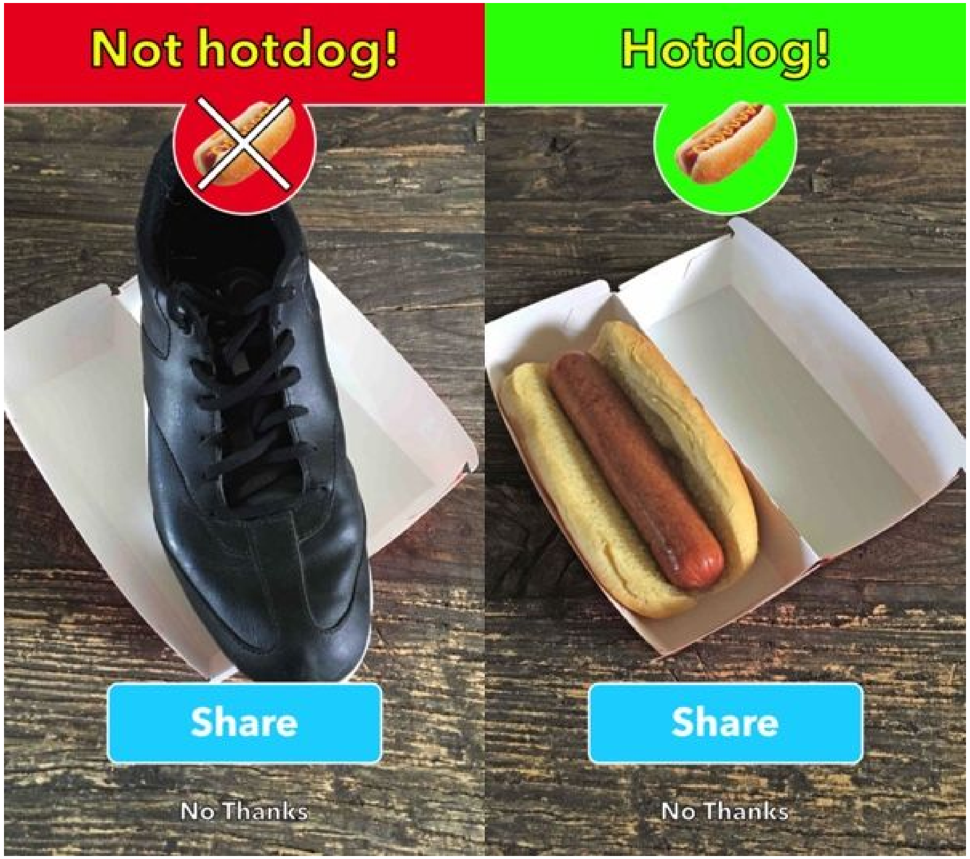
BUILD YOUR OWN SEEK AND DESTROY ROBOT

WHO AM I ?

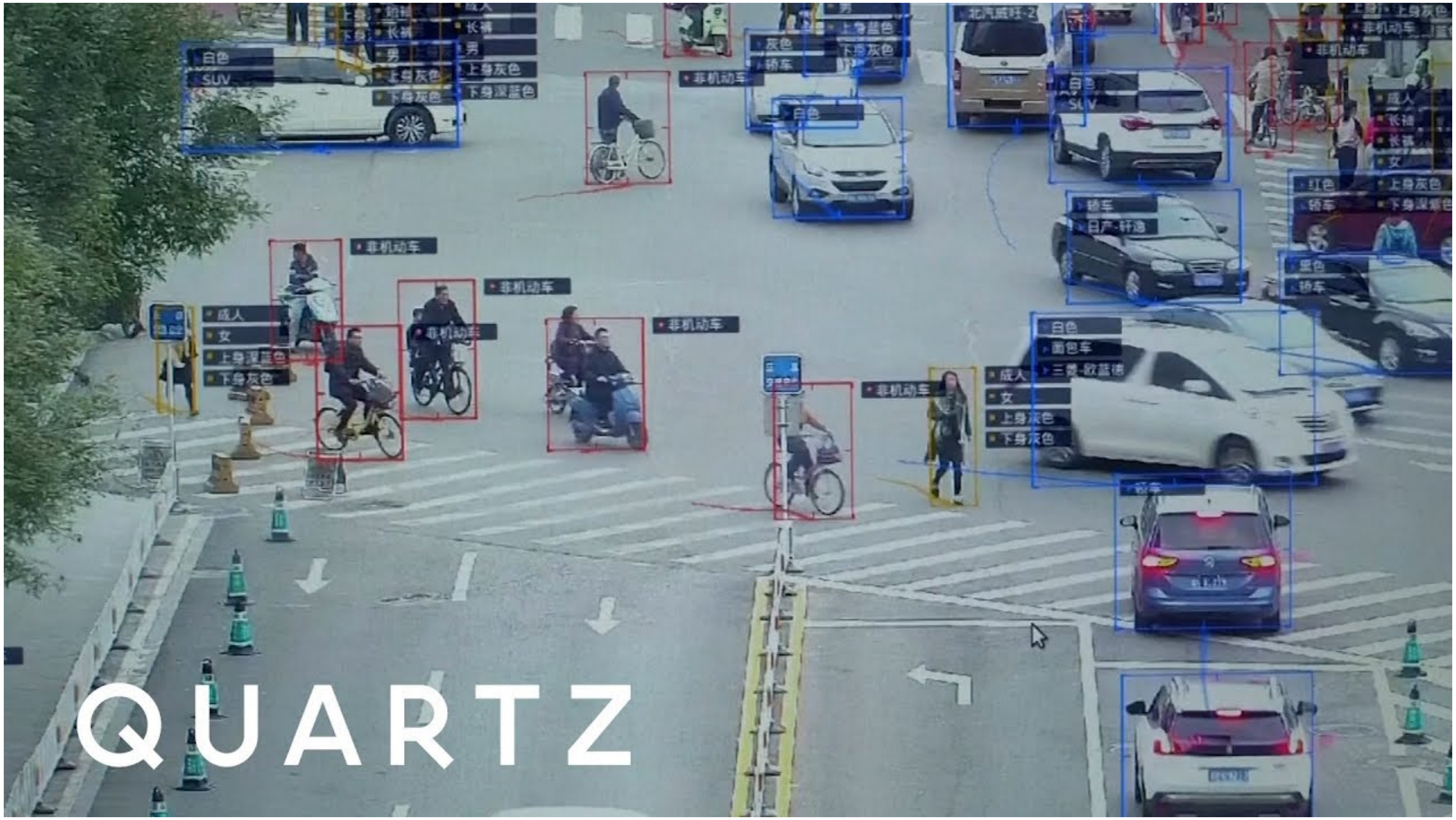
- Senior Security Researcher @ digital.security
- Definitely **not** a ML expert / data scientist
- Love learning new things !

INTRODUCTION

MACHINE LEARNING IS COOL !



LOOKS AWESOME !



DEEPFAKES !

I'M GOING TO LEARN ML

- That's a **challenge** for me
- I have no clue what I'm doing
- Nevermind, **I'll learn** (as usual)

MY LITTLE PROJECT

MY LITTLE PROJECT

- I need to start **small**

MY LITTLE PROJECT

- I need to start **small**
- I need something that will give some **results shortly**

MY LITTLE PROJECT

- I need to start **small**
- I need something that will give some **results shortly**
- Something related to **IoT security**, indeed

digital.security

MY LITTLE PROJECT

- I need to start **small**
- I need something that will give some **results shortly**
- Something related to **IoT security**, indeed
- A tool that gives **a big picture** about IoT ?

DESIRED FEATURES

digital.security

DESIRED FEATURES

- Scans and collect device info from **HTTP** services on **known ports**

DESIRED FEATURES

- Scans and collect device info from **HTTP** services on **known ports**
- Automatically **classifies** these devices

DESIRED FEATURES

- Scans and collect device info from **HTTP** services on **known ports**
- Automatically **classifies** these devices
- Provides an **overview** of customer-premises devices available on the Internet

digital.security

DESIRED FEATURES

- Scans and collect device info from **HTTP** services on **known ports**
- Automatically **classifies** these devices
- Provides an **overview** of customer-premises devices available on the Internet
- Can be used to create **targeted attacks** !

PREVIOUS RESEARCH

- All Things Considered: An Analysis of IoT Devices on Home Networks - **USENIX 2019, Kumar & Al.**
- ProfilIoT: A Machine Learning Approach for IoT Device Identification Based on Network Traffic Analysis - **Yair Medan & Al.**

BUT HOW IS IT DONE ?

BUT HOW IS IT DONE ?

HOW ??

digital.security

MACHINE LEARNING FOR ~~DUMMIES~~ HACKERS

HOW CAN A MACHINE LEARN ?

HOW CAN A MACHINE LEARN ?

THE SAME WAY OUR BRAIN LEARNS.

HOW CAN A MACHINE LEARN ?

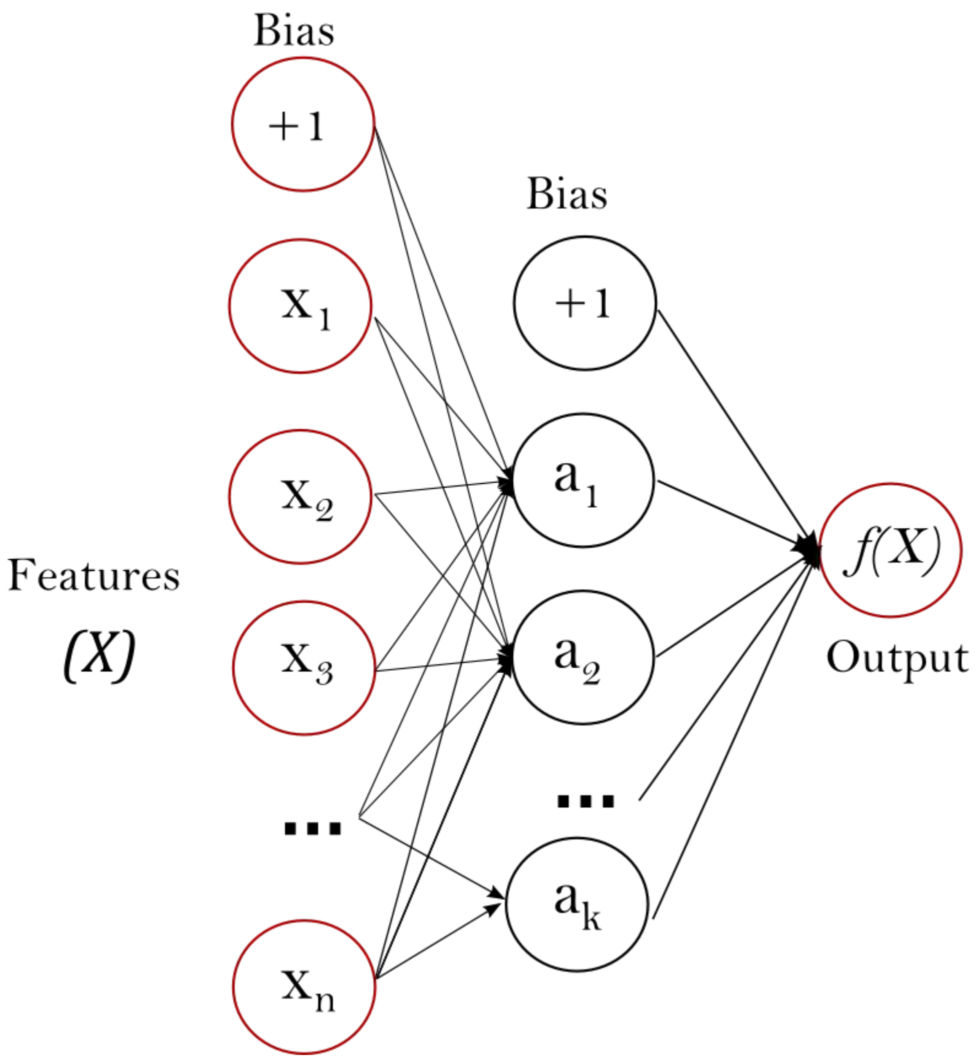
THE SAME WAY OUR BRAIN LEARNS.

(THANKS CAPT'N OBVIOUS...)

TRAIN AND PREDICT

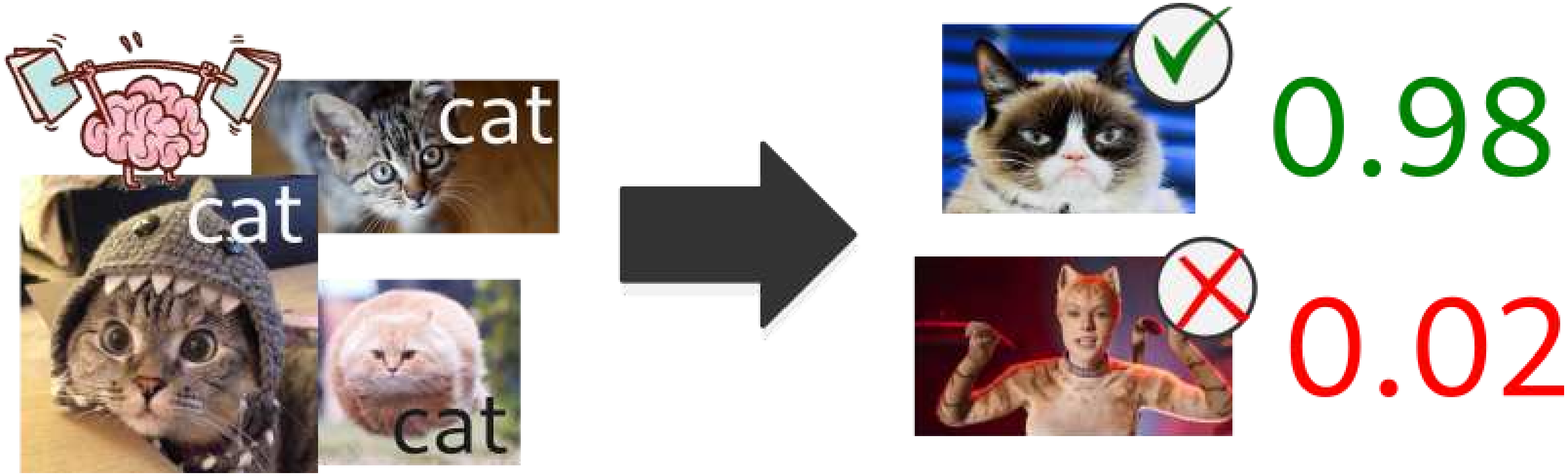
- Train a machine to do a precise task
  (e.g. answer **"is there a cat in this image ?"**)
- Ask the trained machine to **answer the same question** on random images
- This is called **supervised learning**

THE PERCEPTRON



digital.security

TRAIN AND PREDICT



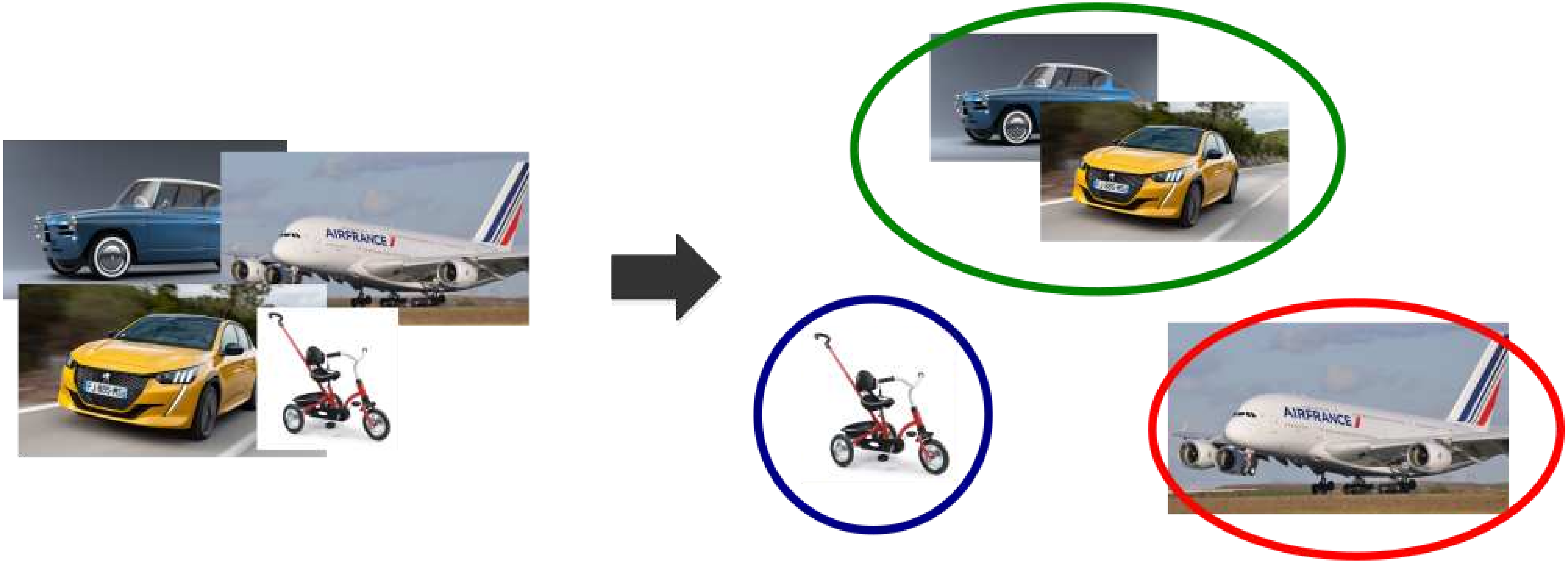
CLASSIFY

- Ask a machine to sort a set of images (e.g. group them by cats, dogs, etc.)
- The machine will find similarities between these images and group them
- This is called **unsupervised learning**

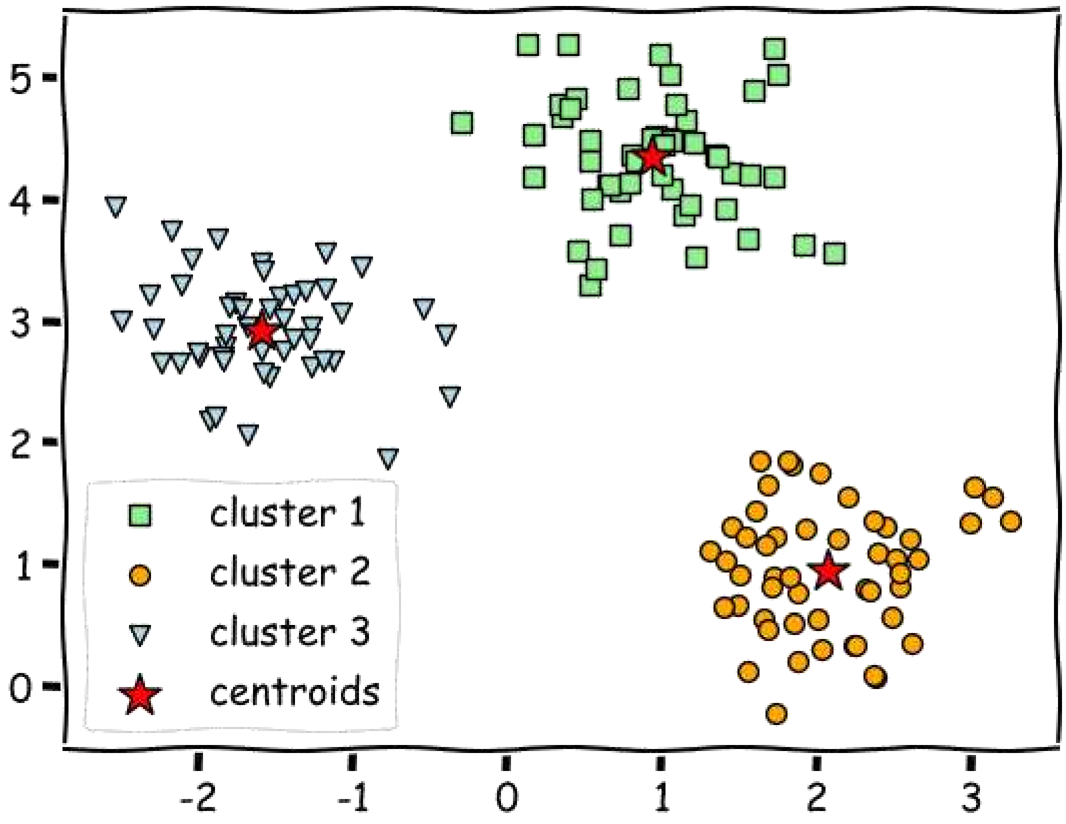
EXAMPLE

- We want to sort a set of data about vehicles
- Describe each vehicle
  - number of wheels
  - number of seats
- Let the machine do the rest !

CLASSIFY



K-MEANS CLUSTERING



K-MEANS CLUSTERING

- Number of centroids (K) is **set at the beginning**
- If K is **too low**, groups will contain **multiple subgroups**
- If K is **too high**, groups will be **spread among multiple centroids**

OTHER ALGORITHMS (WE WON'T COVER)

- **Fuzzy C-means**: similar to K-means but data points are weighted
- **Hierarchical Clustering**

SUPERVISED VS. UNSUPERVISED

- **Supervised learning** is for **training**
  - Two datasets required
  - Training dataset needs associated results set

- **Unsupervised learning** finds **relationships** in chaotic data

SUPERVISED VS. UNSUPERVISED

- **Supervised learning** is a **simple** and **effective** method
- **Unsupervised learning** is more complex and subject to errors

DATASETS

DATASETS

- **Datasets matter**: if not correctly created, could lead to errors
- Datasets may be **biased**
- **Splitting** a dataset in two for training and testing is **not that easy**

FEATURE VECTOR

- **feature**: a **measurable characteristic** of our input data
- **feature vector**: a N-dimension vector containing features

HOW TO TURN DATA INTO A FEATURE VECTOR ?

COLLECTING AND CONVERTING DATA

SCANNING

- **Scan the Internet** for well-known HTTP ports
- Collect **valuable data**
- Turn every collected page into a **feature vector**

CREATING OUR DATASET

- HTTP headers
- HTTP body
- Web page screenshot

digital.security

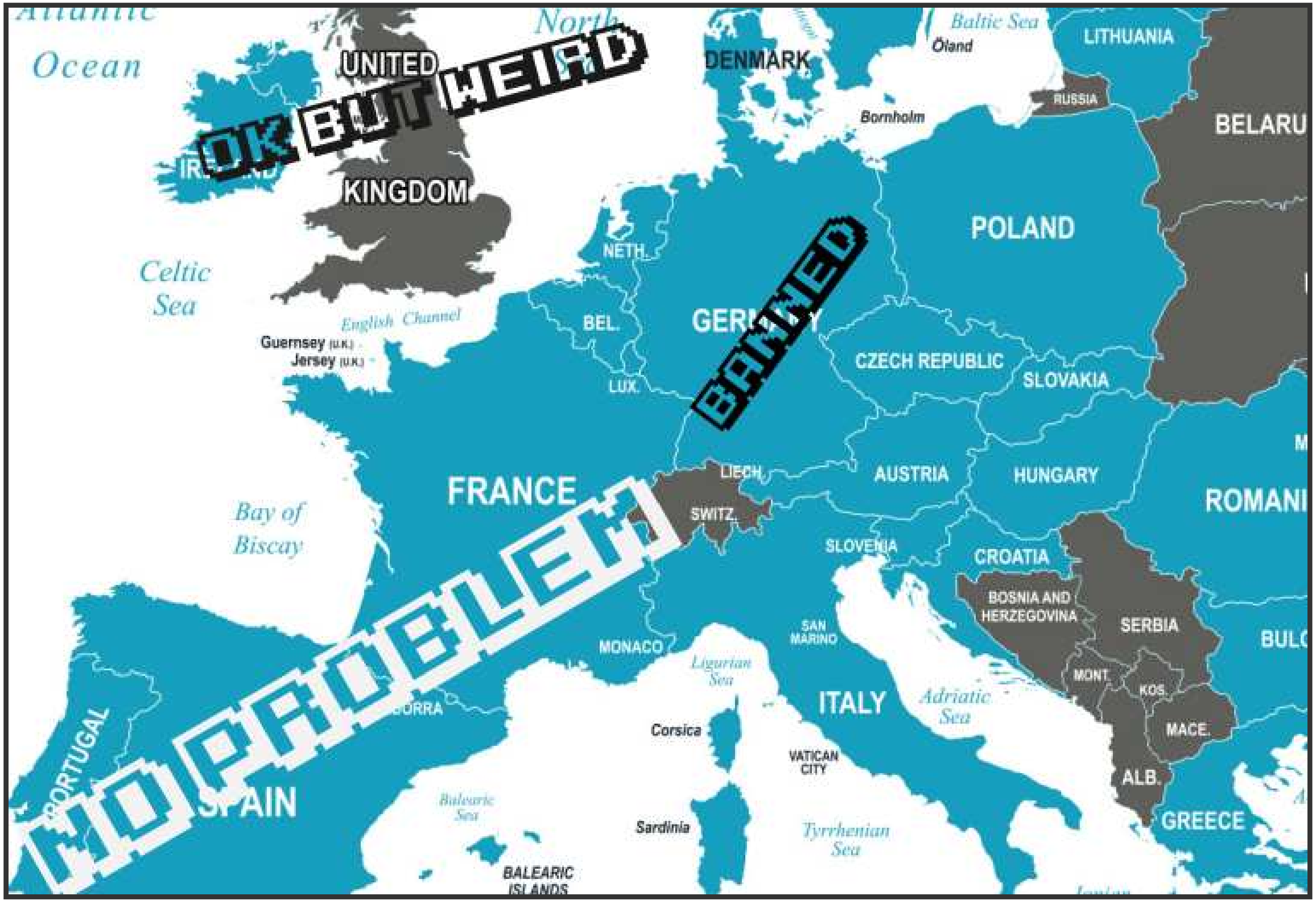
USING REQUESTS TO SCRAPE DATA

```
# Query page
result = requests.get(
  'http://%s:%d/' % (self.ip_address, self.port),
  timeout=1.0
)
headers = json.dumps(dict(result.headers))
body = result.text

# Report target
self.report_target(
  self.ip_address,
  self.port,
  headers,
  body
)
```

CHROMIUM + SELENIUM

```
# Configure Chromium
self.chrome_options = Options()
self.chrome_options.add_argument("--headless")
self.chrome_options.binary_location = '/usr/bin/chromium'
self.driver = webdriver.Chrome(
        chrome_options=self.chrome_options
)
self.driver.set_page_load_timeout(30)
self.driver.fullscreen_window()
# ...

# Save screenshot
self.driver.save_screenshot(dest)
```

ANARCHY IN THE EU



RESULTS

```
$ sqlite3 targets.db
SQLite version 3.27.2 2019-02-25 16:06:06
Enter ".help" for usage hints.
sqlite> select count(*) from targets;
4901
```

digital.security

RESULTS

HOW TO MEASURE A WEB PAGE

HOW TO MEASURE A WEB PAGE

- **content length**: usually the same / device

HOW TO MEASURE A WEB PAGE

- **content length**: usually the same / device
- **number of headers**

HOW TO MEASURE A WEB PAGE

- **content length**: usually the same / device
- **number of headers**
- number of **scripts**, **images** and other **tags**

HOW TO MEASURE A WEB PAGE (BADASS MODE)

- **Levenshtein** distance to a reference page
- **DOM tree structure flattening** combined with Levenshtein distance
- **Normalized** page text size

LEVENSHTEIN DISTANCE (FTR)

- Measures the **difference** between two strings
- Gives a **positive integer** value
- The bigger the value, the bigger the difference

CREATING THE AUTOMATIC CLASSIFIER

SCIKIT-LEARN

- **Python-based** Machine Learning framework
- Built on **NumPy**, **SciPy** and **matplotlib**
- Implements major ML **algorithms**

RECORDS TO DATASET

```
import pandas as pd

def create_dataset_from_records(records):
    """
    Create a ML dataset from a list of records
    """
    lst = [ record_to_values(r) for r in records]
    return pd.DataFrame(lst, columns =[
      'headers','metas','scripts','images','bodysize'
    ])
```

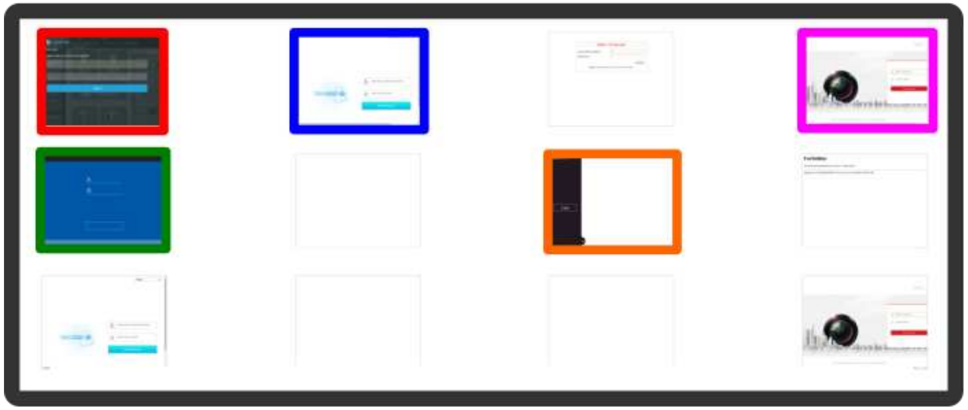
digital.security

IMPLEMENTING K-MEANS

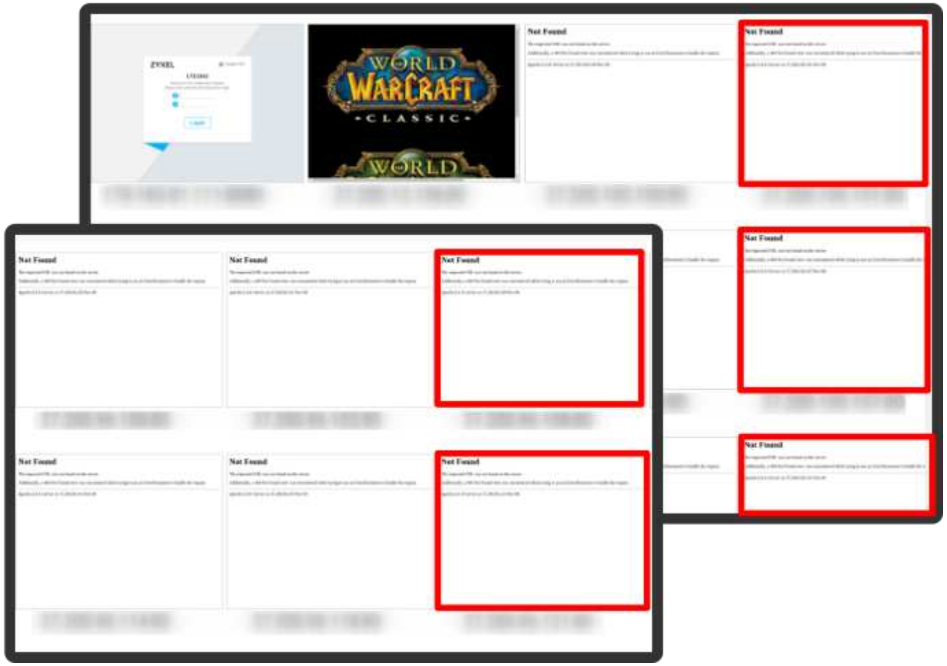
```python
from sklearn.cluster import KMeans
from sklearn import datasets

#...

def classify(records):
    # create a dataset from our DB records
    dataset = create_dataset_from_records(records)

    # classify
    model = KMeans(n_clusters=OPT_CLUSTERS)
    model.fit(dataset)

    # return result
    return model.labels_
```

NUMBER OF CENTROIDS MATTERS

K=100

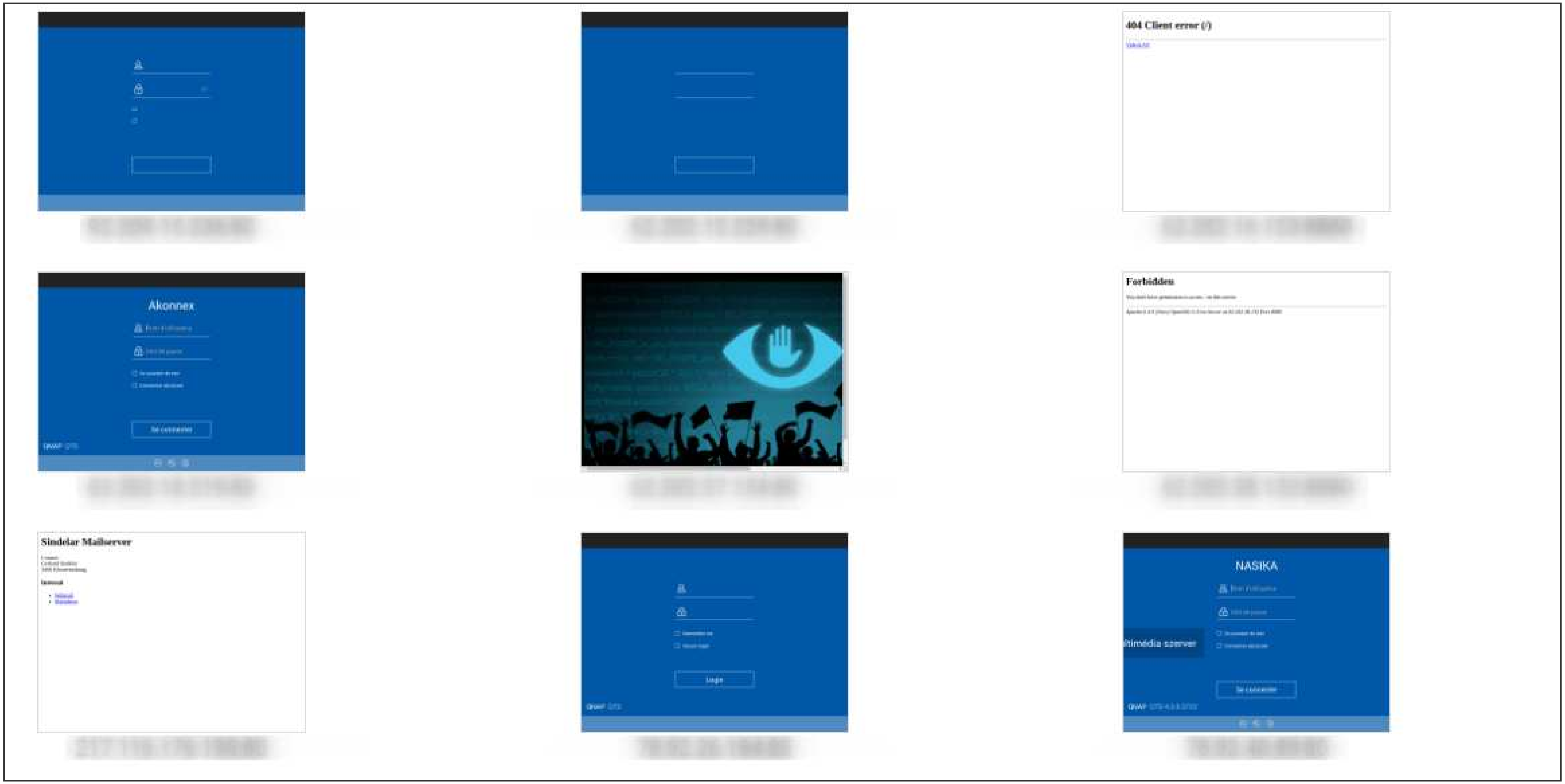


K=1000



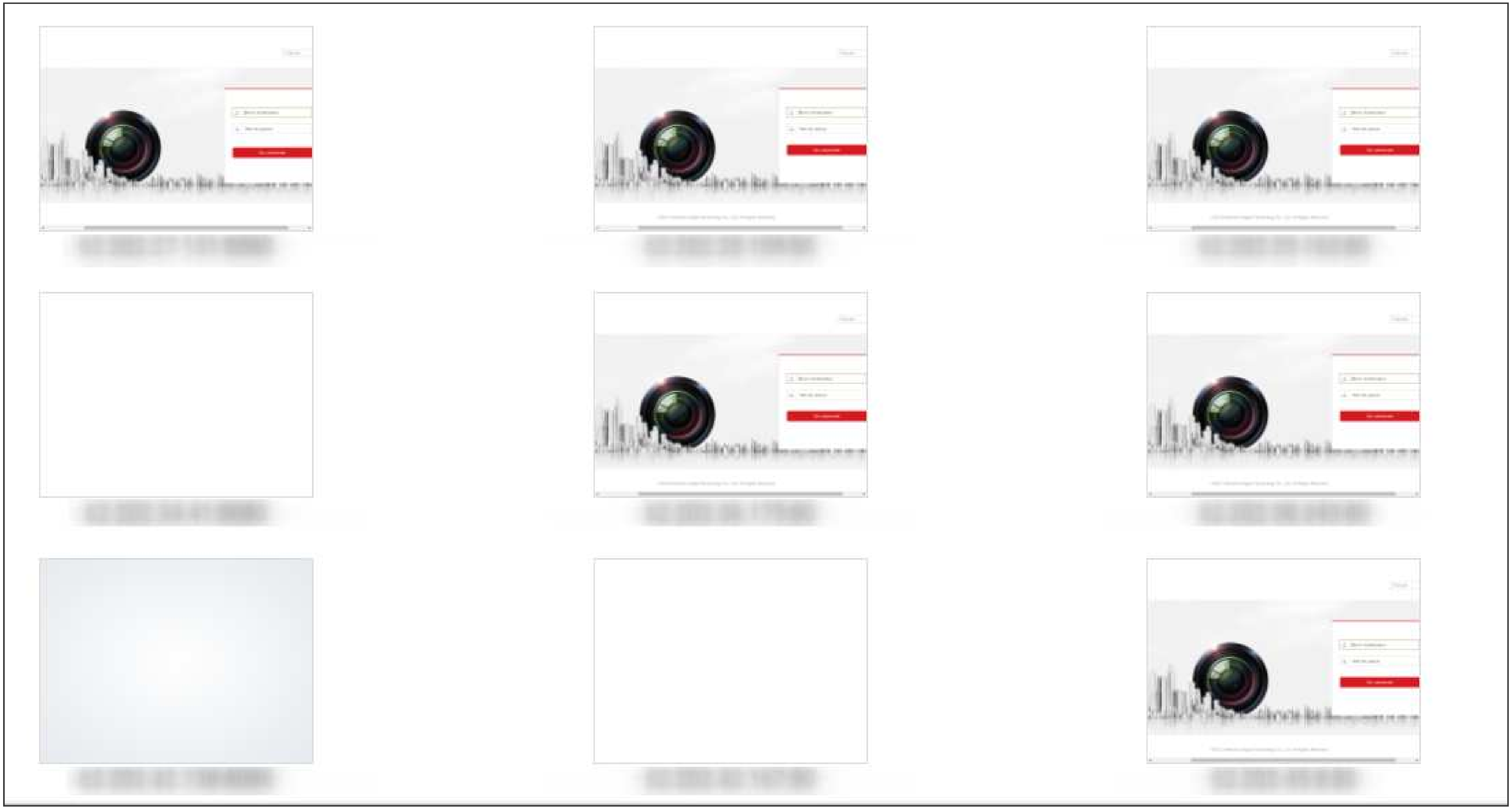
digital security

BADASS FEATURE VECTOR



BASIC FEATURE VECTOR

digital security



BADASS IS NOT THE BEST 😭

- **Levenshtein distance**: two pages with same distance are **not always identical**
- **DOM tree structure**: a lot of devices rely on the **same page structure** (login)
- **Normalized page size**: Most of identical devices have **same content length**

BEST RESULTS 🤯

- 500 centroids
- Content **length**
- Number of various tags (**img**, **meta**, **script**)
- Number of HTTP **headers**

```
4767|213.183.189.11|80|6|1|0|0|120|0.0|0
```

ADDING METADATA

METADATA MAY HELP

- Metada can be useful for **searches**:
  - **category**: NAS, wireless router, etc.
  - **vendor**
  - **product name/series**
- What if we were able to automatically determine (at least) the **category** ?

ML-BASED METADATA

- **Supervised learning**: this is the way.
- We need a **reference dataset** with verified metadata
- Let's add **metadata** to our classified targets !

TRAIN A MODEL FOR EACH CATEGORY

- We create and train a **perceptron** for each category
- We need to have **enough input data** (i.e. targets)

digital.security

PERCEPTRON FOR CAMERA

```
# Collect items from database
targets = list(IotTarget.select())

# Only keep items that ARE cameras
result = [1.0 if (item.category == 'camera') else 0.0
  for item in targets
]

# Build a dataset
dataset = create_dataset_from_records(items)

# Create and train our perceptron
ppn,scaler = create_mlc(dataset, result)
```

digital.security

USING A MULTI-LAYER PERCEPTRON (MLP)

```python
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler

def create_mlc(dataset, resultset):
    """
    Create a multi-layer perceptron (MLP)
    """
    sc = StandardScaler()
    sc.fit(dataset)
    std_dataset = sc.transform(dataset)
    clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
      hidden_layer_sizes=(5,12,15 ),
      random_state=1)
    clf.fit(std_dataset, resultset)
    return (clf, sc)
```

GENERATING A MODEL FOR THIS CATEGORY

```
from joblib import dump

dump((ppn, scaler) , 'camera.model')
```

TESTING THE ACCURACY OF OUR MODEL

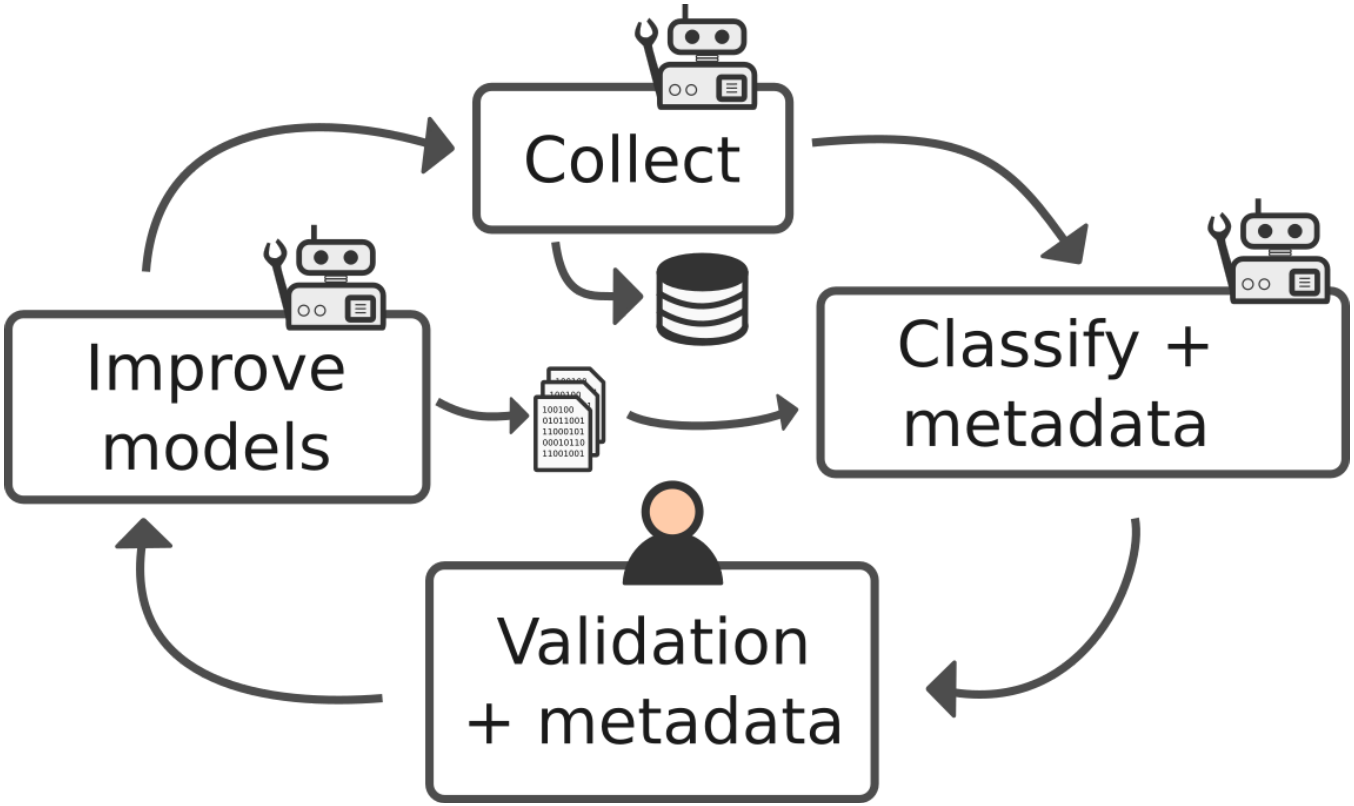
```
t_dataset = scaler.transform(dataset)
y_pred = ppn.predict(t_dataset)
print('Accuracy: %.2f' % accuracy_score(result, y_pred))
```

```
Accuracy: 0.94
```


ReactionGIF.org

digital.security

WORKFLOW



(PARTLY) REVEALING THE IOT LANDSCAPE

SCANNING THE DSL INTERNET

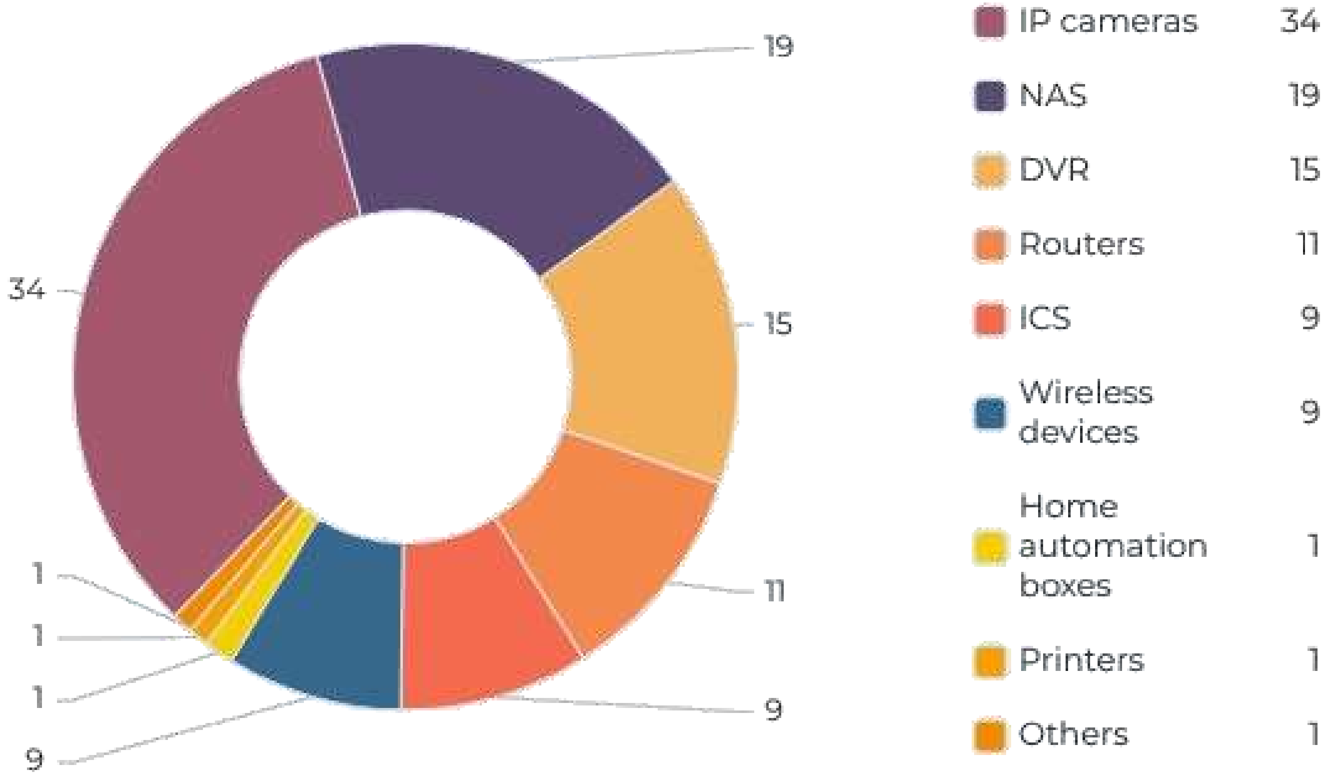
- I discovered almost **5,000 web services** hosted on **DSL IP adresses**
- My tool helped me a lot to **sort this data**
- This is a **small dataset**, but seems **accurate**

RAW DATA

- **4895** web services detected
- **1501** categorized devices
- **3152** screenshots taken
- **34 MB** of HTTP responses content

digital security

IOT DEVICES PER CATEGORY (%)

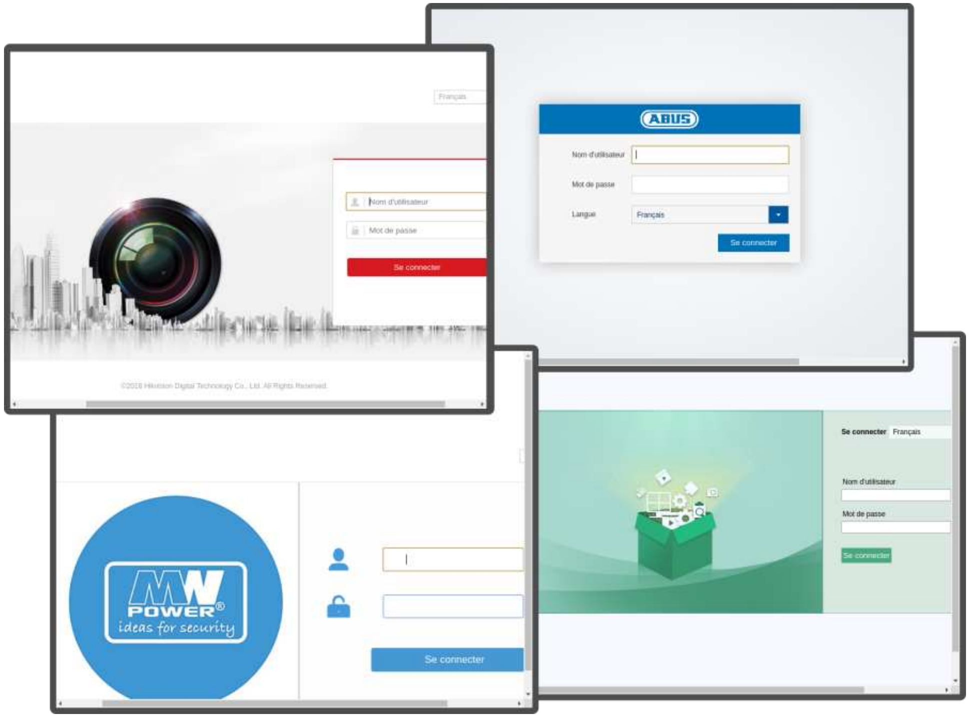


digital.security

TOP 5 VENDORS

#	Vendor	devices
1	Hikvision	372
2	Dahua	117
3	Sonicwall	106
4	TP-link	85
5	Mikrotik	71

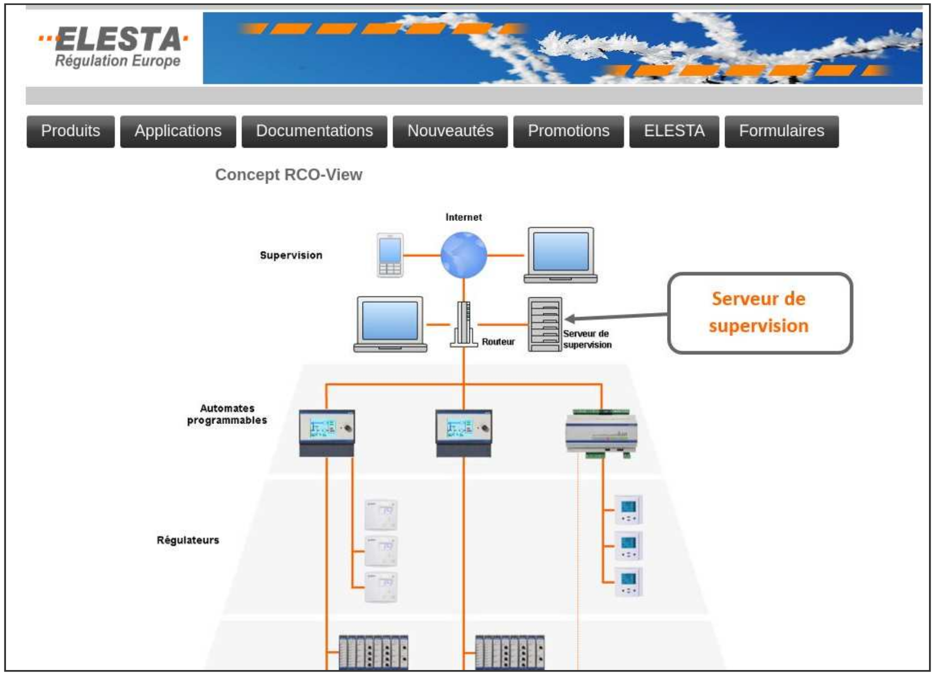
ML IDENTIFIED SIMILAR DEVICES BUT DIFFERENT BRANDS



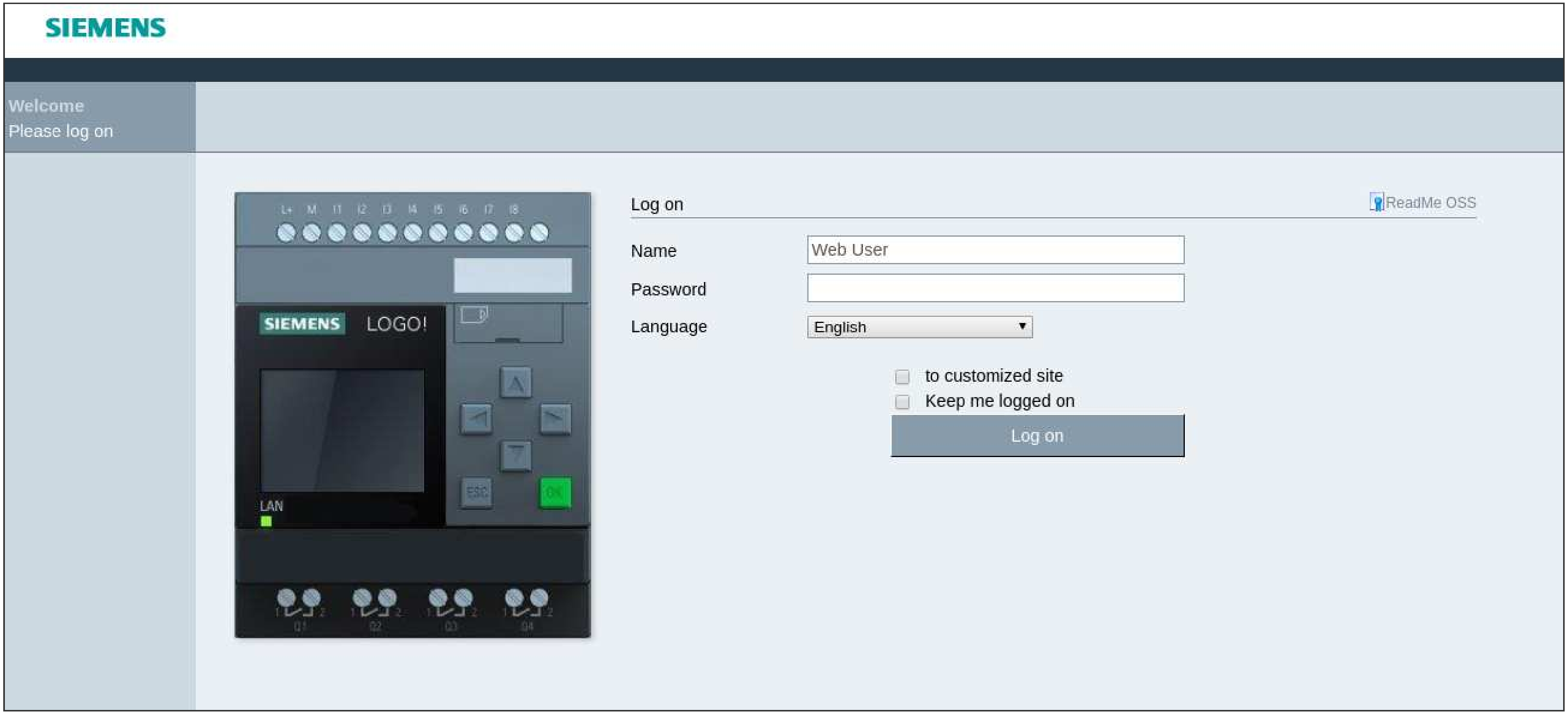
BUT I ALSO FOUND MANY OTHER DEVICES

OT / IT


ELESTA
building automation
Anmelden
Login:
Passwort:
Passwort speichern
Weiter
Projekt wählen

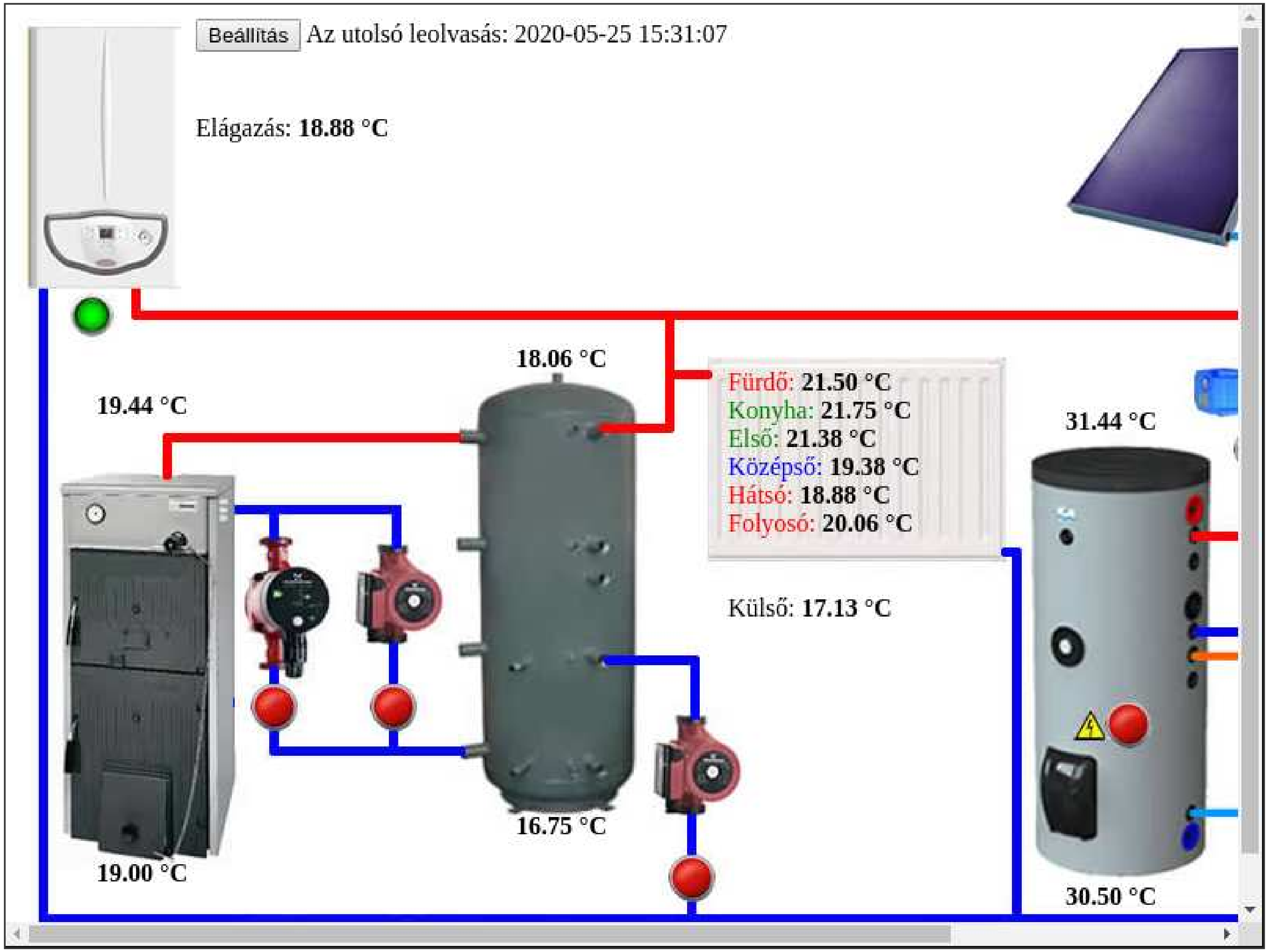

ELESTA
Régulation Europe
Produits
Applications
Documentations
Nouveautés
Promotions
ELESTA
Formulaires
Concept RCO-View
Internet
Supervision
Routeur
Serveur de supervision
Serveur de supervision
Automates programmables
Régulateurs

PRETTY LIABLE CONTROLLER

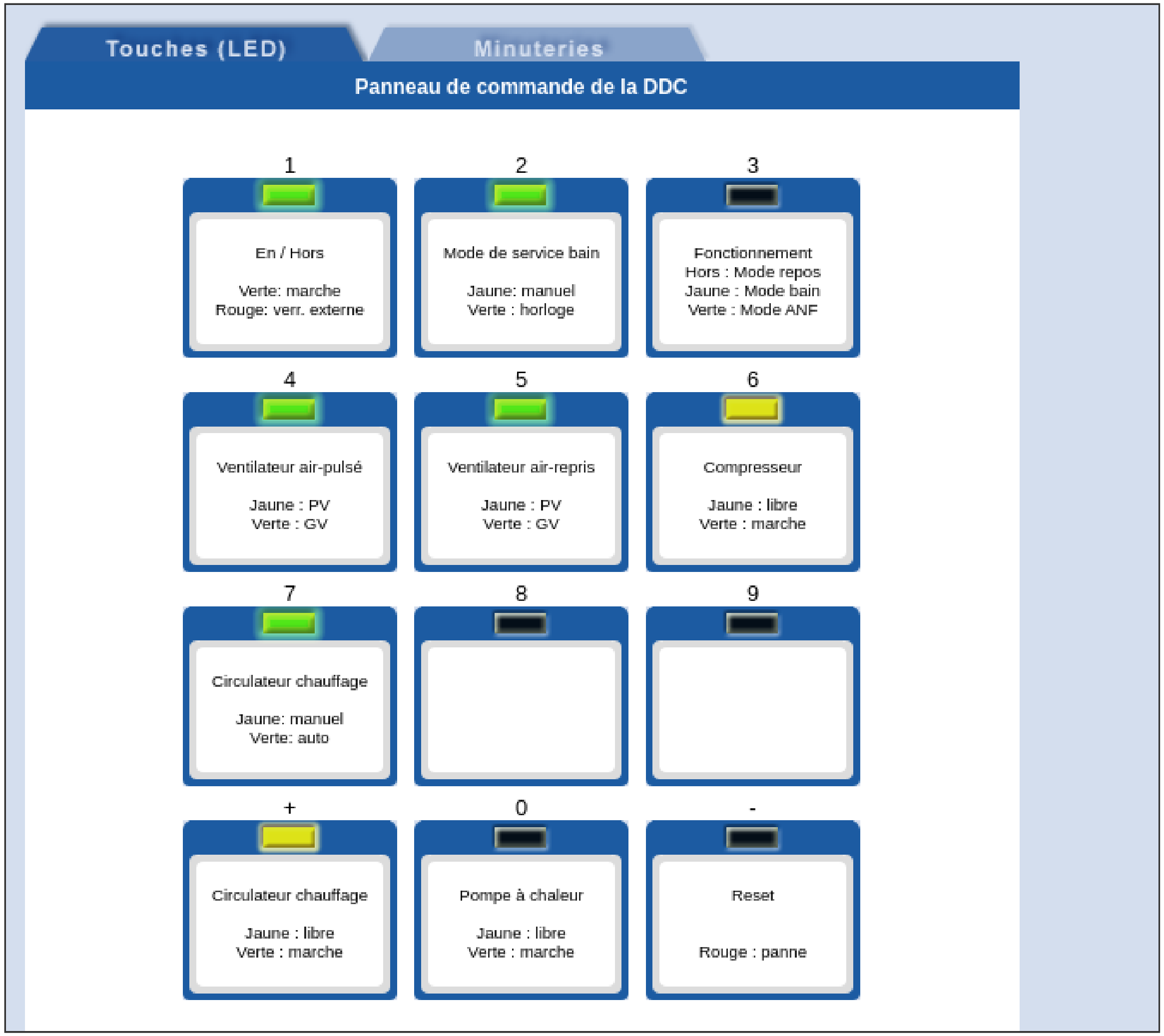


WIND OF CHANGE

WHAT CAN POSSIBLY GO WRONG ?



WANNA SWIM ?


Touches (LED)
Minuteries
Panneau de commande de la DDC
1
En / Hors
Verte: marche
Rouge: verr. externe
2
Mode de service bain
Jaune: manuel
Verte : horloge
3
Fonctionnement
Hors : Mode repos
Jaune : Mode bain
Verte : Mode ANF
4
Ventilateur air-pulsé
Jaune : PV
Verte : GV
5
Ventilateur air-repris
Jaune : PV
Verte : GV
6
Compresseur
Jaune : libre
Verte : marche
7
Circulateur chauffage
Jaune: manuel
Verte: auto
8
9
+
Circulateur chauffage
Jaune : libre
Verte : marche
0
Pompe à chaleur
Jaune : libre
Verte : marche
-
Reset
Rouge : panne

WEAPONIZE

USING ML TO TARGET DEVICES

BUILDING EFFICIENT SCANNERS

- Identifying a **category of devices** is difficult ...
- ... unless you use a trained **perceptron**.

DEMO: SCANNING CAMERAS

GEOLOCATED CAMERA FEEDS

- Identify **camera feeds** (RTP/RTSP) from exposed cameras
- Try **default usernames and passwords**
- **Geolocate** IP address (geoip2)

DOCUMENT THEFT AND RANSOM

- Scan the Internet for **NAS**
- Bruteforce authentication (**default passwords**)
- **Steal data**, leave a note asking for **bitcoins**

SPECIFIC VULNERABILITY RESEARCH AND EXPLOITATION

LOOKING FOR QNAP QTS

- Recent vulnerabilities affecting **QNAP NAS** (pre-auth root RCE)
- It is possible to train a **perceptron** to detect QNAP NAS
- Search & destroy !

CONCLUSION

ML IS GREAT

- **Unsupervised classifier** allows quick devices review
- **Multi-layer perceptron** provides an easy way to create **targeted tools**, assign **metadata**
- **Human** is still required !

TAKEAWAYS

- Machine learning algorithms are easy to use with **scikit-learn** and Python
- Extra libraries required: **requests**, **whoosh**, **sqlite3**
- The **most difficult** part: picking the **right features** and building correct **datasets**

I WON'T RELEASE ANY SOURCE CODE

- All the **key material** is provided (code snippets, parameters, etc.)

- I **learned a lot** during this project, so will you 👍

- Well, maybe because **my code is dirty** too...

THANK YOU FOR LISTENING

HOPE YOU ENJOYED THE TALK 😁

DISCOVER NEW THINGS, EXPERIMENT, LEARN !

Twitter: @virtualabs

damien.cauquil@digital.security