Anomaly detection and explanation in networks with machine learning

Maxime Lanvin, **Pierre-François Gimenez**, Ludovic Mé, Yufei Han, Éric Totel, Frédéric Majorczyk
CentraleSupélec, Inria, IMT (Télécom SudParis), DGA

NICT/INRIA/IMT seminar, March 2nd, 2023

CentraleSupélec

Context

Context of this work

- Work on a network intrusion detection system that monitors network packets
- Anomaly detection: we modelize legitimate behavior based on benign training data with no access to attacks
- Based on Sec2graph by a previous PhD (Laetitia Leichtnam)

Goals

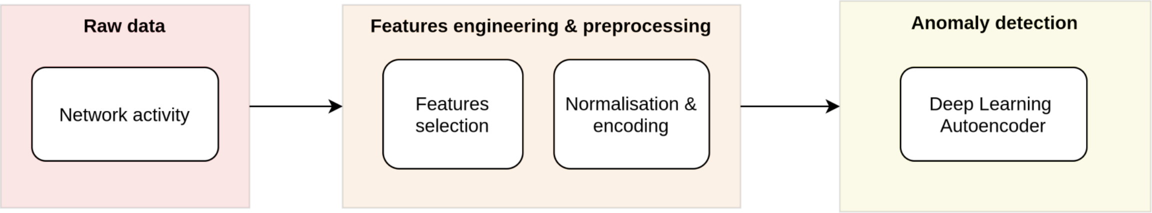
- Have good detection performances with limited false positives
- Provide explanations for alerts with a new XAI mechanism

Outline

Overview of the approach

Structure of the approach

- Probes capture the network data
- These data are merged into a graph structure
- The graph is transformed into a format usable with a deep learning model
- The model affects an anomaly score to each data point



Data capture

CentraleSupélec

Probe

For now, we rely on public datasets, most notably:

- CIC-IDS2017 (5 days of traffic, 14 machines)
- CSE-CIC-IDS2018 (several weeks, 500 machines)
- DAPT2020 (5 days, 5 machines)

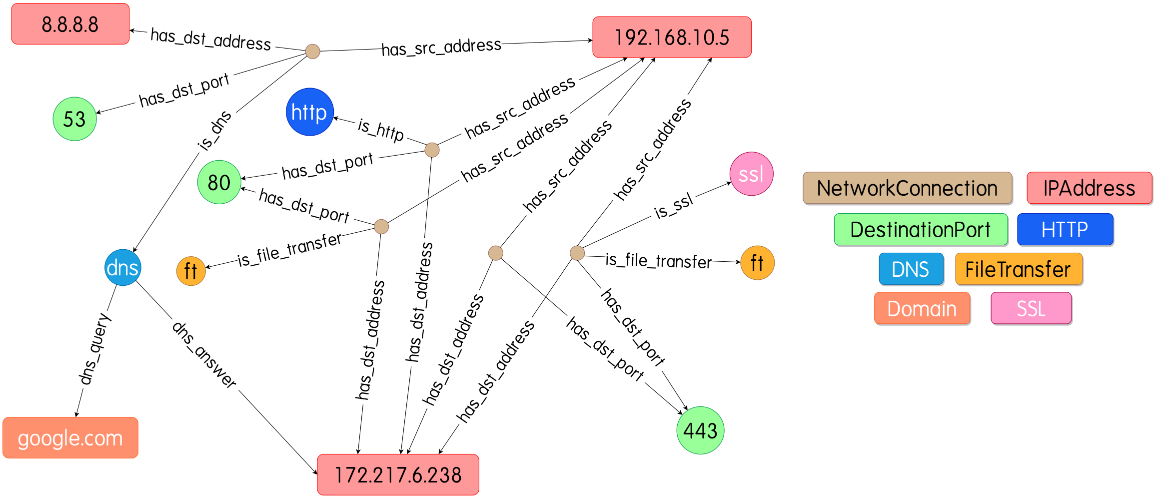
We work directly on the pcap files (the raw capture) and not on the higher levels features

Packet dissector

- We use Zeek (formerly Bro) to dissect the packets
- Zeek creates multiple log files, one for each category of events (network connection, HTTP request, x509 certificates, etc.)
- All events are associated with one network connection

Next step: construct a graph from these logs

Security objects graph built from Zeek's logs



Security objects graph

Nodes

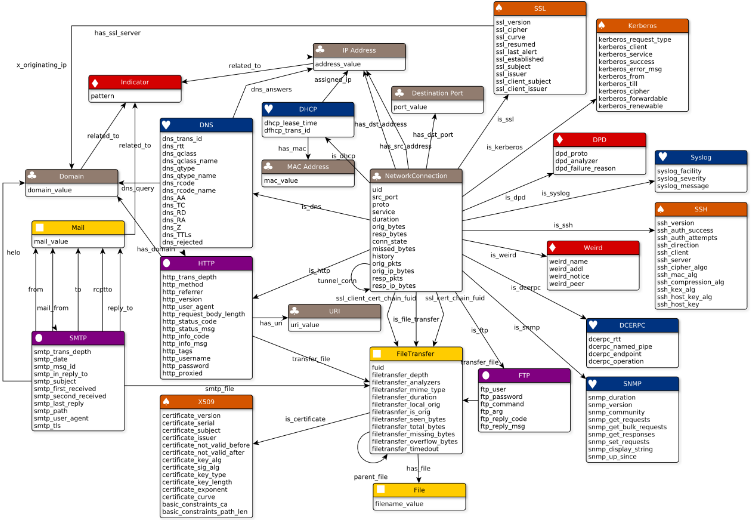
- Each node type corresponds to a "security object":
  - protocols: DNS, SSH, DCERPC, SNMP, FTP, DHCP, HTTP, SMTP
  - network data: port, MAC address, IP address, network connection, URI, domain
  - and others
- Nodes contain a set of attributes related to these objects

Edges

- Edges are typed and oriented
- An edge between two nodes means that these two nodes are found within the same event

CentraleSupélec

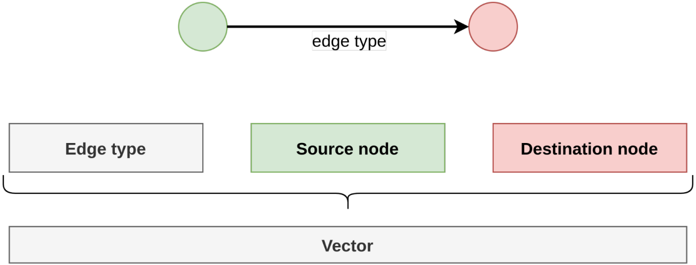
All nodes and edge types



Graph encoding

Why is the issue?

- We cannot feed the model with the whole graph $\Rightarrow$ we process the graph edge by edge
- Deep learning models generally require a fixed-sized vector with numerical values:
  - To encode discrete values (like port number or protocol), we use one-hot encoding (one feature per value)
  - To encode continuous values (like connection duration), we use a Gaussian mixture model

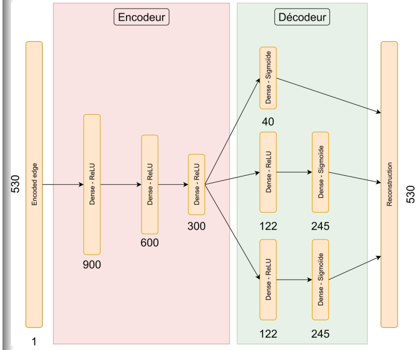


Edge encoding scheme

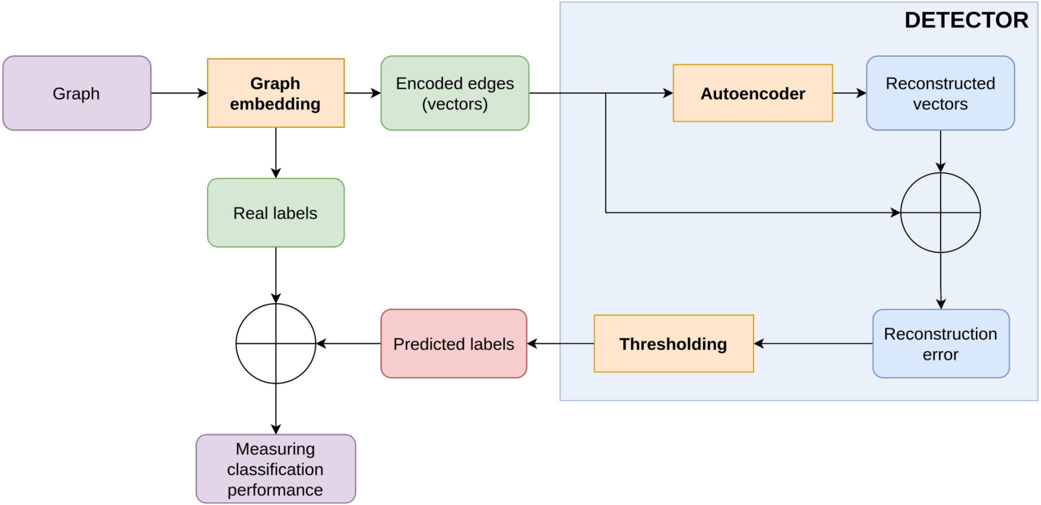
Deep learning model: autoencoder

Autoencoder

- An autoencoder is a deep learning model with the shape of a bow tie
- During the learning phase, the model tries to reconstruct its input data as faithfully as possible
- Due to the bow tie structure, the model needs to find a way to compress the input data by learning the underlying structure of the data
- Once learned, the model is very effective at reconstructing inputs that resemble the training data
- But the compression fails on data too different from the training data!
- We use the reconstruction error as an anomaly score



Summary



DETECTOR
Graph
Graph embedding
Encoded edges (vectors)
Autoencoder
Reconstructed vectors
Real labels
Predicted labels
Thresholding
Reconstruction error
Measuring classification performance

Performances on DAPT2020

CentraleSupélec

Performances

- Experiment on DAPT2020 dataset with APT attacks
- Comparison with the best unsupervised solution proposed by the article (SAE)
- Sec2graph is almost always better
- It has a good recall (it correctly identifies a lot of attacks) and a reasonable false positive rate. However, it's not mature yet for real-world application: it still has too many false positives

	AUC ROC		AUC PR	
APT attack step	*SAE*	*Sec2graph*	*SAE*	*Sec2graph*
Reconnaissance	0.641	**0.888**	0.262	**0.613**
Foothold Establishment	0.846	**0.924**	**0.498**	0.480
Lateral movement	0.634	**0.802**	0.014	**0.603**

Our next goal was to study false positives to reduce them

How to explain the predictions?

The issue

- Motivation: explanations could help us understand:
  - the false positives, to enhance our detector
  - the true positives, to provide SOC experts more information about the alert
- There exists a lot of explanation techniques: LIME, SHAP, salient maps, counterfactual explanation...
- ...but little work on explanations for anomaly detection!

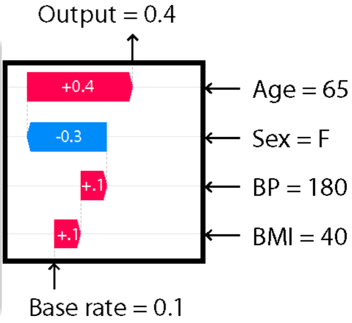
Evaluated methods

- SHAP has been used successfully with autoencoder
- A more naive approach found in the literature
- A new explanation method we propose for reconstruction-based anomaly detection

State-of-the-art methods

SHAP

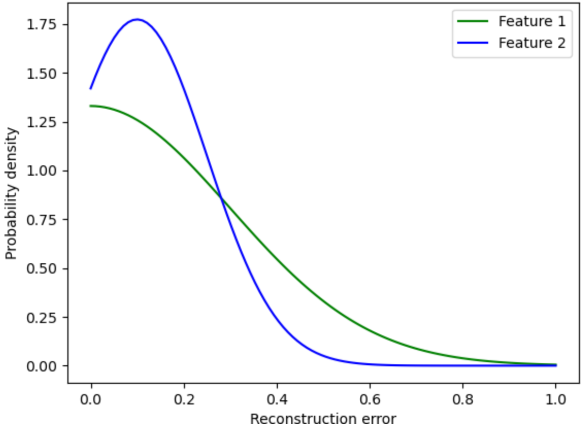
- A black-box local XAI method, based on a theoretical framework (shapley values)
- It finds the contribution of each features to the output
- It has been used for autoencoder but more adapted to classification
- Drawback: very slow



Naive approach

- An approach used in the literature before the XAI methods
- Reminder: anomaly typically have high reconstruction error
- We can check what feature contribute the most to this reconstruction error
- The bigger the reconstruction error, the more important the feature is

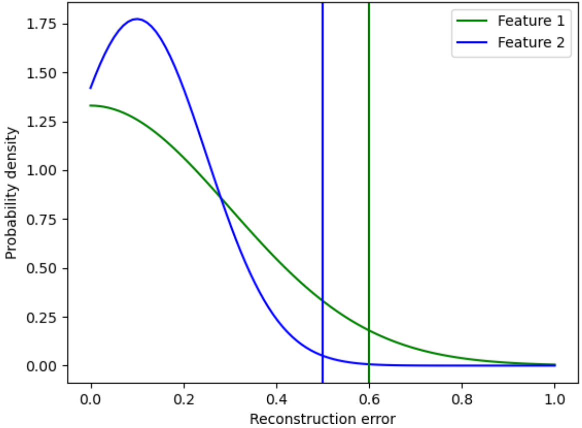
Naive approach



However, this idea does not always produce satisfactory explanations:

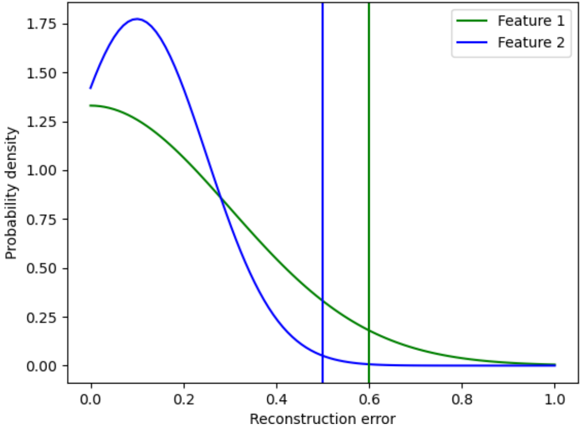
- Some feature have a high variance in the dataset, hence they are difficult to reconstruct (example: the client port)
- Some features are closely correlated to other features, making them easy to reconstruct, and even a small reconstruction error may reveal an anomaly (example: the transport protocol, UDP/TCP/ICMP, is highly correlated with the application protocol, the server port, etc.)

Example

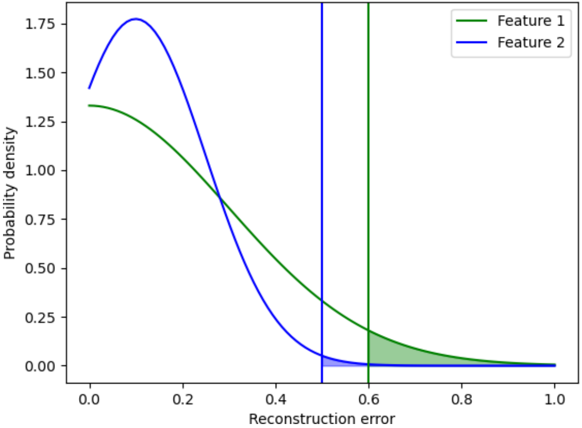


Naive method: feature 1 is the explanation because the reconstruction error is higher

Example



Naive method: feature 1 is the explanation because the reconstruction error is higher



But an error of 0.5 is rarer for feature 1 than an error of 0.6 for feature 2...

Our XAI method

Our proposal: a statistical explanation

First step: calibration

- With training data, we estimate the distribution of reconstruction error of each feature

Second step: explanation

- We compare the reconstruction error of each feature to these distributions
- The explanation is the feature with the lowest p-value of the reconstruction error

Some remarks

- Advantage: we can deal with features that can be either hard or easy to reconstruct
- Drawback: we require some training data

Experimental protocol

CentraleSupélec

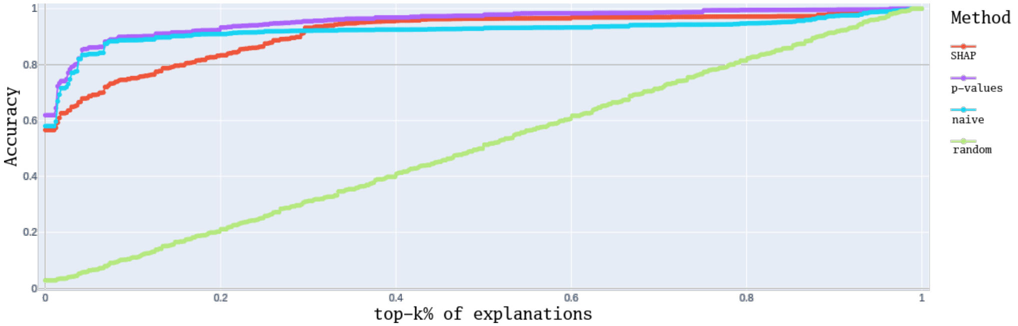
Experiment 1 protocol: identify the inserted anomaly

- We select a vector from the network data
- We modify the value of a random feature (it's the inserted anomaly)
- We generate the explanation for this modified vector
- We evaluate the accuracy of the explanation: did the method find the location of the anomaly?

Experiment 2 protocol: use case (CIC-IDS2017)

- We used our method to generate explanations for attacks and false positive
- We merge explanations of several edges related to the same network connection
- We compare the explanations with the expert's knowledge

First experiment: accuracy



Results

- Our method is the most accurate explainer, followed by the naive explainer and SHAP
- 80% of the time, the modified dimension is within the top 5% explanations for p-values
- With other enhancements, the gap between naive method and our method gets wider

First experiment: time

Method	Time per explanation
Naive method	1.0 ms
Our method	1.9 ms
SHAP	28.41 s

Time comparison

- The naive method is very fast: it just a maximum search
- Our method is more complex but with a proper implementation it is very fast
- SHAP is really slow, as expected

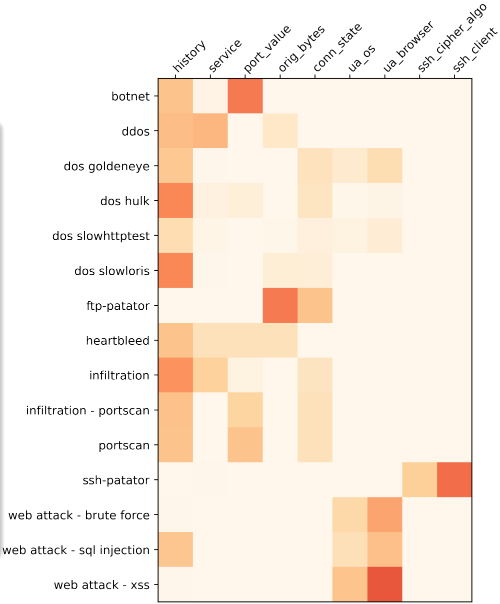
Second experiment: use case

What about real attacks?

We used the explainer on real attacks detected by our IDS. There are some clear correlations between attacks and features:

- TCP flag history is useful to detect many attacks
- SSH bruteforce is visible by the SSH client feature
- Botnet and portscan use atypical server ports
- Web attack are launched from tools with atypical user agents

Still a work in progress, but it is promising



Second experiment: use case

False positive in CIC-IDS2017

The explanations told us where the problem was... in the dataset itself!

- Labeling issues: a port scan attack (70,000 flows) was incorrectly labeled as "benign"!
- Duplication issue: a badly configured probe duplicated data in the dataset (500,000 extra packets per day)
- A few more minors issues

Why wasn't it found before?

Turns out that the missing attack has duplicated packets, so its csv files didn't look like the other scan attacks. Consequence: supervised methods miss this unlabeled attack

We published a fixed version of the CICFlowMeter tool and the dataset

Conclusion and future work

CentraleSupélec

Conclusion

- An anomaly detection approach relying on a security objects graph
- Our new XAI method is accurate, fast, and can be used with any reconstructed-based anomaly detection (as long as we have access to calibration data)
- It shows promising results on false and true positives, but it still a work in progress

Future work

- Edges should not be processed independently: embeddings and attention mechanisms could help exploit the neighborhood
- Time series analysis is crucial for APT detection: we plan to add new edges between network connections in the security objects graph, with a temporal semantics
- The security graph objects could be extended with other data sources, e.g., application logs