Some work of starting PhDs in CIDRE on AI and cybersecurity

Pierre-François Gimenez
CentraleSupélec, IRISA

GT stats seminar, June 9th, 2022

Context

AI expertise is a new skill in the team

- CIDRE is an Inria team focused on cybersecurity
- Two new members were recruited with a background in AI and cybersecurity
  - Pierre-François Gimenez, Maître de Conférences CentraleSupélec in 2020
  - Yufei Han, ARP Inria in 2021

Four new PhDs started in October 2021

- Maxime Lanvin* on intrusion detection
- Adrien Schoen* on data generation
- Vincent Raulin* on malware analysis
- Hélène Orsini on botnet detection

* I am one of their supervisors

This presentation is a brief overview of their work

CentraleSupélec

Work of Maxime Lanvin

Supervisors

Ludovic Mé, Yufei Han, Pierre-François Gimenez, Éric Totel (Télécom SudParis), Frédéric Majorczyk (DGA)

Context

- Work on a network intrusion detection system that monitors network packets
- Anomaly detection: we model legitimate behavior based on benign training data
- Based on Sec2graph by a previous PhD (Laetitia Leichtnam)

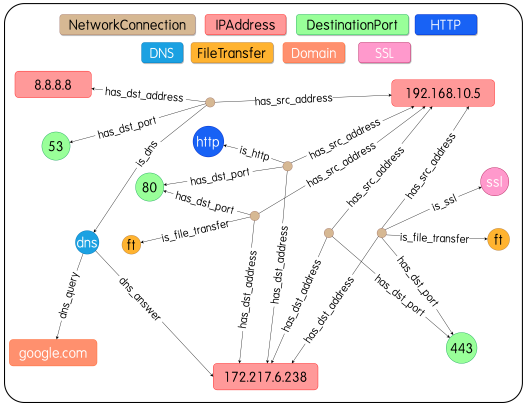
Goals

- Provide explanations for alerts
- Limit false positives
- Detect complex APT (Advanced Persistent Threat) attacks

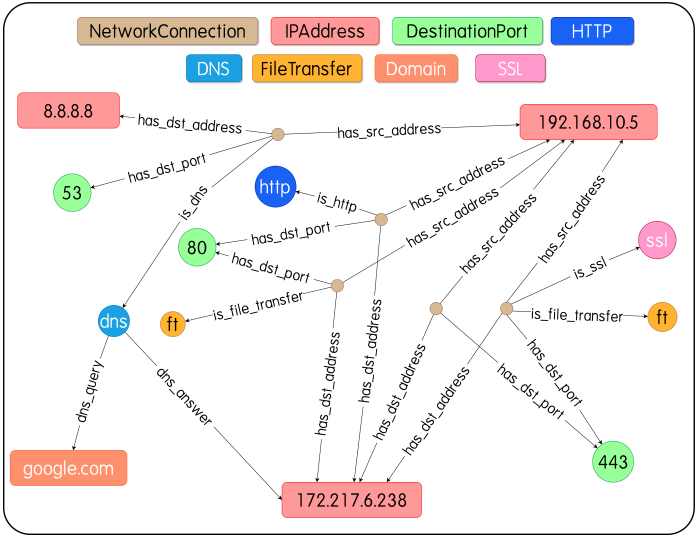
Approach: Sec2graph

Approach (based on Sec2graph)

- Gather packets from a network
- Build a security objects graph
- Encode each edge of the graph into a vector
- Learn an autoencoder to reconstruct each edge of the graph from benign data
- During inference:
  - Reconstruct the input with the autoencoder
  - Measure the reconstruction error
  - Raise an alert when it exceeds a threshold



Security object graph



Performances

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 good recall (it correctly identify a lot of attacks) and reasonable false positive rate. But it has some limits!

	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**

CentraleSupélec

Limits of the method

Very local analysis of the graph

- Each edge is processed independently
- $\Rightarrow$ Embeddings and attention mechanisms could help exploiting the neighborhood

No time analysis

- Long-term, discrete attacks (APT) could evade the detector
- $\Rightarrow$ graphs generally hide the temporal dependencies between objects

No explanation

- We know which edges have high anomaly score, but we don't know why
- $\Rightarrow$ Very few explanation methods for unsupervised learning

We focus on explanations for now

CentraleSupélec

Unsupervised explanation

The issue

- We had a lot of false positives on the dataset that we could not understand
- There exist a lot of explanation techniques... (LIME, salient maps, counterfactual explanation...)
- ...but little work on explanation for unsupervised learning!

Our approach

- For each edge, we have the input vector and the output (reconstructed) vector
- First idea: compute the error feature-wise. However, some feature are harder to reconstruct than others, so the explanation is very noisy!
- Second idea: modelize the distribution of reconstruction error per feature

Statistical explanation

Our current approach

- We learn the model with benign data
- We compute reconstruction error of another set of benign data
- For each feature, we estimate its distribution of reconstruction error
- During inference, we use the product of the p-value of the reconstruction error for each feature
- The detection threshold is based on this p-value
- It is really easy to isolate the contribution of each feature and output the most influential features to an expert

Results

- Our detection performances got a little bit better
- It allowed us to find serious labeling issues in the dataset we were using!

CentraleSupélec

Conclusion

Article on the problematic dataset

- An attack was not correctly labeled
- After the fix, our method has less false positive...
- ...and supervised methods from literature have worse results!
- We believe supervised models of the state of the art were overfitted

Upcoming work

- Enhance the explanation method (maybe with the help of a statistician?)
- Apply the method to industrial protocols as well
- Integrate embeddings or time models

CentraleSupélec

Work of Adrien Schoen

Supervisors

Ludovic Mé, Yufei Han, P.F. Gimenez, Grégory Blanc (TSP), Frédéric Majorczyk (DGA)

Context

- Network intrusion detection systems (NIDS) monitor network packets
- To evaluate these NIDSes, we need benign and malicious network traffic
- Malicious traffic can be generated with dedicated tools
- Due to class imbalance, we need a lot more benign traffic

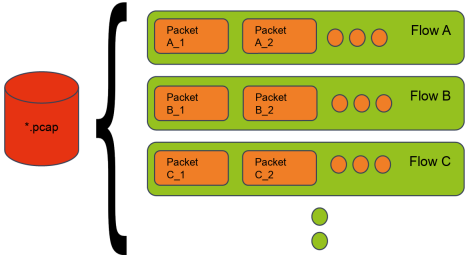
How to get benign traffic

- Record it from a real network $\rightarrow$ privacy issues, tedious experiment, obsolescence, etc.
- Record it from a simulated network $\rightarrow$ requires modeling and simulating end users, etc.
- Generate it with statistical and machine learning methods $\rightarrow$ our approach

First challenge: generation

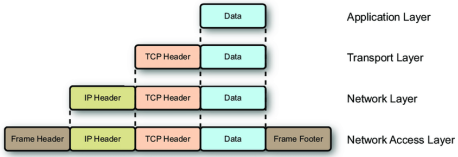
Different scales

- There are different scales of data in network packets
  - A pcap file is a list of packets
  - Each packet is part of a flow (a connection)
  - Flows are not independent from each other
  - Each packet is composed of encapsulated protocol headers and a payload



Different issues

- Network packets flows are time-series,
- Header complies with network protocols,
- Payload is generally encrypted
- Etc.



Generation methods

Statistical methods

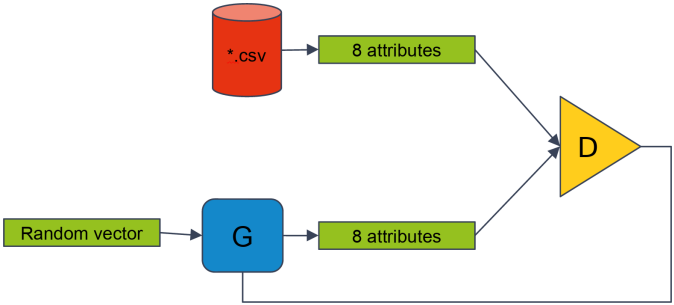
Bayesian networks are probabilistic graphical models

- Advantages: explainable model, easy to train, fast generation (+ a dynamic version for time series generation)
- Disadvantages: few implementations, notably for both discrete and numeric features. Theoretical issues with determinism (faithfulness assumption not satisfied)

Deep learning models

- Variational autoencoder (VAE)
  - Advantages: not too complex,
  - Disadvantages: not the best performances, hard to explain
- Generative adversarial network (GAN) $\rightarrow$ what we are using right now
  - Advantages: good performances
  - Disadvantages: difficult to train (two models), mode collapse, hard to explain

What does our generation look like?



	Duration	Proto	Src IP Addr	Src Pt	Dst IP Addr	Dst Pt	Packets	Bytes
0	0.000921	TCP	192.168.220.15	80	192.168.220.15	80	2	54
1	0.002552	TCP	192.168.200.9	40289	96.76.60.29	39151	4	1581
2	0.000017	TCP	192.168.220.4	443	192.168.220.6	445	3	740

Here is an example of a flow description generation

- We use an embedding for some discrete data such as IP address, source port, protocol, etc.
- Generation is done by a GAN
- The generation is not perfect: source port and destination port are not coherent
- This asks a question: how to evaluate the generation?

CentraleSupélec

Second challenge: evaluation

Criteria and scoring functions

We distinguish two notions:

- Evaluation criteria are very general properties of the generated data
- Scoring functions are functions that assess one or more criteria

Evaluation criteria

- Realism: synthetic sample should belong in the real distribution
- Diversity: all relevant parts of the real distribution should be generable
- Originality: new samples should be different from samples of the training set

A fourth criteria

Network data are discrete (not like pictures), so we need a fourth criteria:

- Compliance: the network sample should conform to protocols specifications

CentraleSupélec

Scoring functions

Scoring functions survey

- There is no consensus yet on how to evaluate synthetic network data
- Many generation work evaluate only some criteria (compliance for example)
- We did a survey of scoring functions from various domain:
  - tabular (vector) data
  - image
  - textual data
  - network traffic
- We expect to use several functions depending on the criteria and the layer
  - textual scores are adapted to temporal series like packet flow
  - tabular scores are adapted to headers/flow description generation
  - image scores are not easily adaptable
  - network traffic scores could evaluate whole pcap files

CentraleSupélec

Scoring functions examples

Data type	Scoring functions	Input	Realism	Diversity	Originality	Comp.
Tabular	Recall, Density	Distribution		✓		
	Precision, Coverage	Distribution	✓			
	Authenticity	Sample			✓	
Image	Inception Score, FID	Distribution	✓	✓		
	MiFID	Distribution	✓	✓	✓	
Textual	BLEU, ROUGE	Sample	✓			
	WMD	Sample	✓	✓		
	BERTscore	Distribution	✓	✓		
	SelfBLEU	Distribution		✓		
Network traffic	DKC, PcapGAN test	Sample				✓
	PAC-GAN test	Sample				✓
	GvR	Distribution	✓	✓		

Conclusion

Upcoming work: evaluation

- We didn't decide yet what scoring functions to use for which scales
- Next step will be implementation

Upcoming work: generation

Top-down approach: we chose to generate from packet flow to their actual content

- We are fixing the flow description generation
- Next step is packet stream generation
- And finally payloads

CentraleSupélec

Work of Vincent Raulin

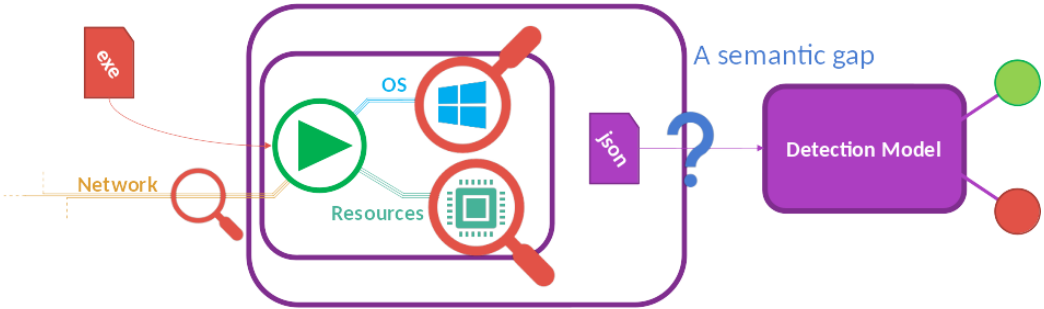
Supervisors

Pierre-François Gimenez, Valérie Viet Triem Tong, Yufei Han

Context

- The malware threat is ever-growing
- Automatic analyzers are required to help the experts keep up with the pace
- There are two main categories of malware analysis:
  - Static analysis: the binary file is analyzed
  - Dynamic analysis: the malware is executed and monitored
- We focus on dynamic analysis, where it's more difficult for the malware to hide its malicious activities

State of the art



The state of the art focuses on:

- source of information, i.e. what to monitor (malware experts)
- statistical and machine learning models to detect, classify and cluster malware (stats/AI experts)

CentraleSupélec

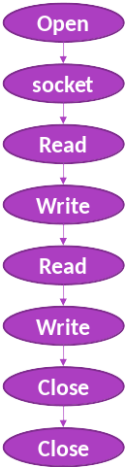
What is missing?

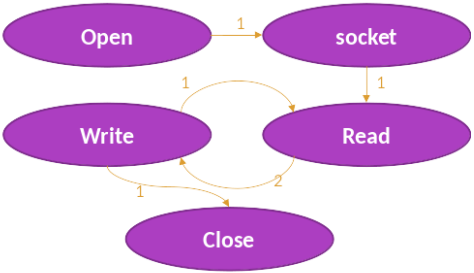
Little work on representation

- How data is represented can have a huge impact on model performances (cf. Maxime's work with security objects graphs)
- Dynamic analysis data is generally low-level and noisy: it's difficult to understand the behavior of a malware from such a trace
- Models better work with high-level data, which less objects but more structure
- There is currently a semantic gap between low-level data and high-level model
- Creating such a representation requires both malware and model expertise

Our first step was to survey the state of the art of dynamic trace representation

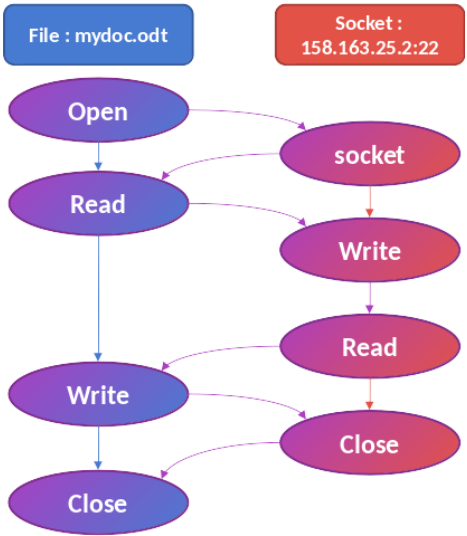
State of the art of dynamic trace representation





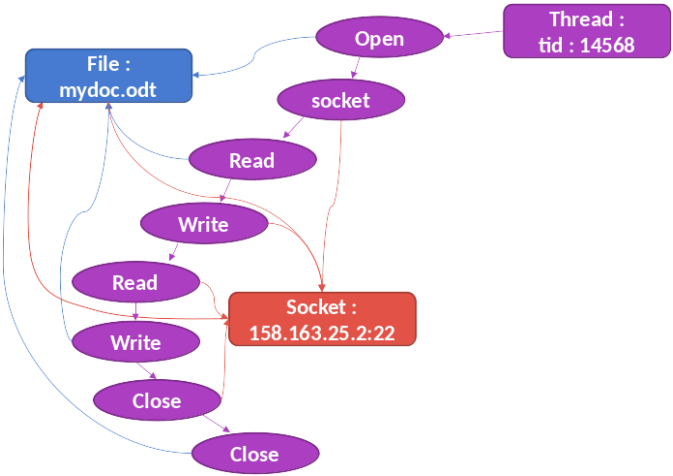
- Sequences of system calls (no parameters)
- Transition frequencies between system calls or groups of calls
- Cannot convey enough information
- Easily subject to adversarial attacks
- Behavior difficult to distinguish

State of the art of dynamic trace representation



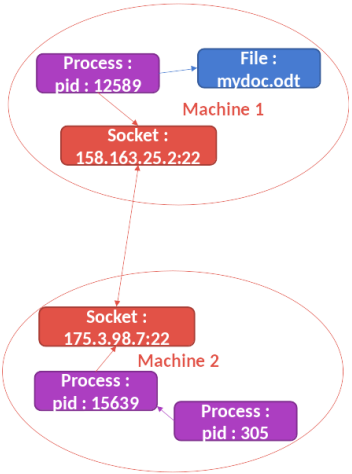
- Different system resources can be seen as object that can be included in some way.
- One sequence of actions per object
- Links between actions on same or related objects
- Still low level representation

State of the art of dynamic trace representation



- Show the objects and the actions on them
- Allows to see the links between objects
- Shows the information flows which convey program behavior
- Too large for experts
- Sample-centric

State of the art of dynamic trace representation



- Shows which processes start which ones
- Shows what machines communicate with each other
- Show the accessed files
- Local behavior is absent
- Monitoring of an entire network
- Cannot be done with our experimental setup

CentraleSupélec

Research questions

Our research questions

- How to make a representation more robust to evading techniques?
  - by removing some noise, it's more difficult for an attacker to modify the representation of the trace of a malware
- How to make a representation abstract enough to make it cross-platform?
  - require only using high-level objects (socket, file, threads, etc.) and no OS-dependent values (system calls, etc.)
- What elements make the representation of an execution trace visually exploitable for a human expert?
  - We will need to summarize a potentially huge representation to a manageable size
- How to ensure a representation reflects all the malicious actions of a malware infection event?
  - Malware creators may find a way to abuse OS quirks and evade the representation

Conclusion

Upcoming work

- A big chunk of time was dedicated to the experimental pipeline (Web crawler, VM preparation, Cuckoo analysis, etc.) but it's now ready for use
- We are still working on our representation. It includes network usage and we are working on file system usage
- Once we have a first version, we will use a simple model to compare it with a baseline representation, and work iteratively from there

CentraleSupélec

Work of Hélène Orsini

Supervisors

Yufei Han, Valérie Viet Triem Tong, David Lubicz (DGA)

Context

- A lot of methods have been used so far for botnet detection: statistics over network flows, aggregation on a time window, graph-based, ...
- Theses methods generally require a lot of parameters set empirically, requiring both experts and time
- There is a loss of information when using overall statistics and aggregation
- Auto-ML could alleviate such problems

Goals

- Machine Learning-driven Network Traffic Flow based Intrusion Detection System (IDS)

CentraleSupélec

Methodology

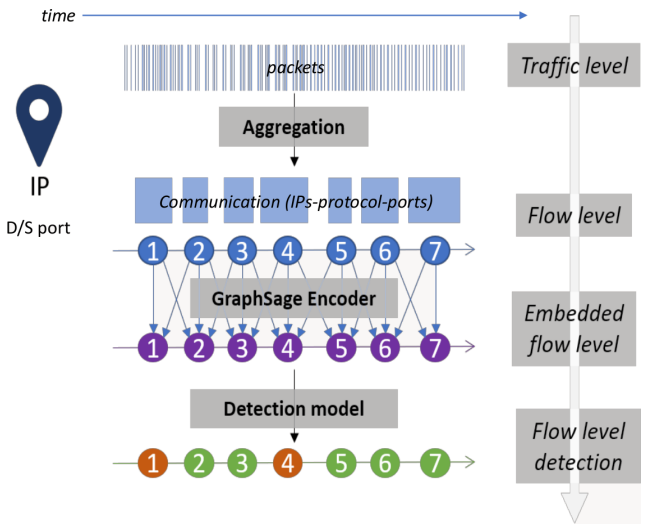
Technology used

- Auto-ML: avoiding feature engineering, learn directly from raw categorical / numerical data
- Transferable ML (Meta-ML): you can reapply your detection model across different botnet traffic datasets without retraining efforts / much retuning efforts
- Explainability: you can figure out which factors / which features trigger the detection / improve transparency of the detection model

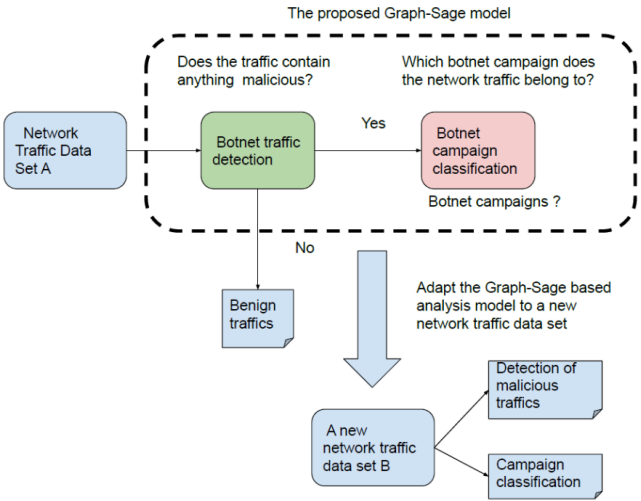
Proposal

Approach based on GraphSage that takes into account the context of each communication

GraphSage method



Proposal and next steps



Expected contributions

- Unsupervised method
- No need to set up parameters (AutoML)
- Adaptability (i.e., no need to retrain totally)
- Explainability

Next steps

- Botnet traffic detection: adapt to another dataset
- Improve botnet classification
- Continue the state of the art

CentraleSupélec

Talk conclusion

Conclusion

- We tackle various domains of security with AI
- These 4 PhD students started 8 months ago and should be ready to publish in a good conference by the end of 2022
- Our research in AI&Cyber goes beyond what these PhD students do
- We want to create collaborations between security experts and stats/ML experts

The AI&Cyber research in CIDRE is new but thriving!