Proof Of Liveness

1/ Plan

-2 march → 9 march (1 week) : Research and Setup
-9 march → 21 march (1.5 week) : Practice and preps for prototype build (Research on tech tools Python libraries, Cairo, useful firmwares…).

***** Upload PrepDoc*****

-21 march → 4 april (2 weeks) : Cahier des charges (And bibliographie)

***** Upload Cahier des charges *****

-4 april → 11 april (1 week) : Conception document

***** Upload Doc de Conc *****

-11 april → 2 may (3,4 weeks) : Local Prototype blocks

*****Upload V1*****

-9 may → 16 may (2 weeks) : Tests, and Updates

*****Upload Final V*****

######## HALF TIME ######## (2 weeks lifeboat)

-30 may → OnChain migration (2 months)
- 1 august → RAPPORT (1 month)

2/ Ideas

- Machine Learning algorithm that detects bots from real humans.
- Neural network that distinguishes human faces from “other” pictures.
- Human mouse trajectory recognition.
- Completely decentralized algorithm (Open source) *not rely, or as less as possible on using Security by obscurity.*
- Not worrying about the robustness of the algorithm*, the goal is to discourage a certain threshold of bots. Simply adding an extra task already removes (personal estimation) 50% of bots. Machine learning algorithms repulce up to 80%, 90% of most advanced bots. There must be a real motivation and dedication to make a powerful intelligent bot, but then, the danger of employing real human farms exists also. Then the effort needed to develop a 100% bot repellent algorithm, seems unnecessary. Complementary document verification is necessary for highly critical systems.*

- Preserving privacy as much as possible.
- Integration de starkware en live.
- Proof of livenes: Is 100% robustness even necessary? After a difficulty level, easier to hire real humans.

3/ Project Summary

4/ Meetings

Nicolas Perrin-Gilbert : perrin@isir.upmc.fr ISIR Researcher (Friday 04/03/22 at 17h):

- Proof of livenes by Machine Learning, (Without document verification), that is completely robust, **Doesn't exist** to date. Big companies protect themselves by using abnormally huge amounts of data to train their Machine Learning algorithms and that way ensuring that very few teams in the world have that much data to train on. For example Facebook is able to do training on user mouse trajectory on Millions of subjects per day. That way having access to the top quality of data which cant be attained in normal conditions.
- **Warning**: Keep in mind that for successful training, a very large data set is required.
- Proof of livenes is generally not based on machine learning, because it's not completely robust. A more **dynamic** solution is required: Changing the nature of the test before there is a breach. The Goal is to make the task of breaching the test by a bot as hard as possible before it has the time to train himself to resolve the test. Nevertheless We have to keep in mind that the test is never 100% impenetrable.
- **Particular problem** of supervised learning: Algorithms must classify a face and "something else".Training by giving to the "else" category random pictures is dangerous. ML algorithms have pretty unpredictable responses to unseen data. To Check: GAN (Adversarial NN). Generator and discriminant methods.
- **Warring**: Security by obscurity, big companies hide in top secrecy their trained neural network.
- **Decentralization**, and open sourcenes gives the project an interesting aspect as we must avoid using hiding anything.

5/ Bibliographie

[1] - Nicolas Perrin-Gilbert, course on supervised learning

Exemple of architecture for the MNIST number classification problem: Pixels as input neurons, and 10 output neurons. The softmax function then transforms the output values in a probability distribution. Entrainement: Iteratively, we pick random batches from the train data set. The minimisation of the negative log likelihood is a cross entropy minimization between the real probability distribution and the model. Gradient descent to do the minimisation. Backpropagation

is a particular way of doing that for neural networks. This simple architecture can attain 90% of good classification on the MNIST database.
Convolutions: Adding filter layers to the input images.
Max-Pooling:A problem with the output feature maps is that they are sensitive to the location of the features in the input. One approach to address this sensitivity is to down sample the feature maps. Operation that calculates the maximum value in each patch of each feature map. Highlights the most present feature in the patch, not the average presence of the feature in the case of average pooling.
Dropout: Technique used to prevent overfitting. Dropout works by randomly setting the outgoing edges of hidden units (neurons that make up hidden layers) to 0 at each update of the training phase.
GANs: Given a training set, this technique learns to generate new data with the same statistics as the training set. The generator is not trained to minimize the distance to a specific image, but rather to fool the discriminator.

[2] - Gunjan, V. K., Senatore, S., Kumar, A., Gao, X. Z., & Merugu, S. (Éds.). (2020). Advances in Cybernetics, Cognition, and Machine Learning for Communication Technologies. Lecture Notes in Electrical Engineering. https://doi.org/10.1007/978-981-15-3125-5

Artificial Neural Network and Partial Pattern Recognition to Detect Malware : Signature bases and behavior based Malware detection, heuristic based (works on unknown malware as well). Methodology used: Opcode extraction, Feature selection, Feature vector construction, Classifier architecture, Training, Testing. An unknown malware detection system that is automated. OpCode extraction was done with 3-gram model. ANN classification of malware families almost with accuracy of 100%.

Chapter 8: lkj

[3] - Meng Joo Er, Shiqian Wu, Juwei Lu, & Hock Lye Toh. (2002). Face recognition with radial basis function (RBF) neural networks. *IEEE Transactions on Neural Networks*, *13*(3), 697-710. https://doi.org/10.1109/tnn.2002.1000134

lkk

[4] - Lior Goldberg Shahar Papini Michael Riabzev. (2021) Cairo – a Turing-complete STARK-friendly CPU architecture.

[5] - Alejandro Acien , Aythami Morales , Julian Fierrez, Ruben Vera-Rodriguez. (2021). BeCAPTCHA-Mouse: Synthetic Mouse Trajectories and Improved Bot Detection. https://arxiv.org/pdf/2005.00890.pdf

lkj