# Deep in the Dark - Deep Learning-based Malware Traffic Detection without Expert Knowledge

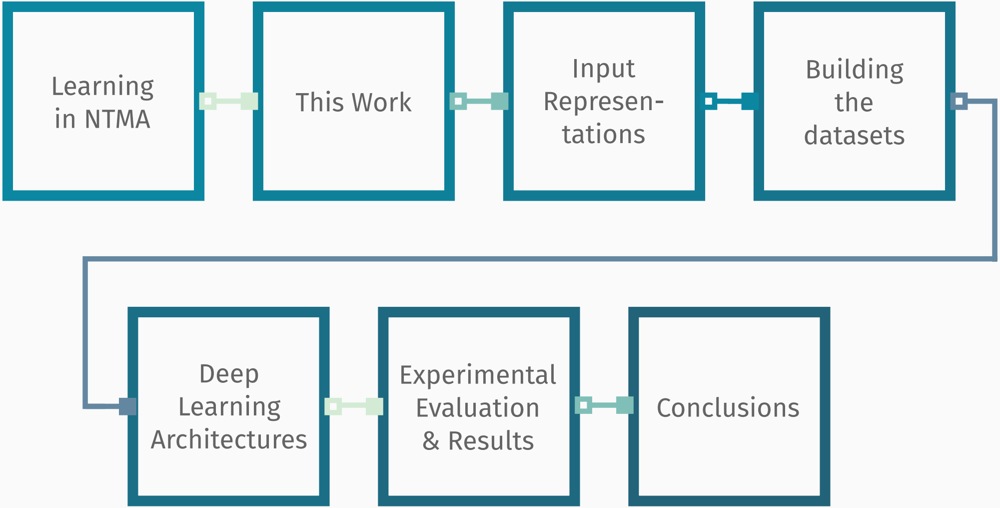
Gonzalo Marín[1,2]   Germán Capdehourat[2]   Pedro Casas[3]

(1) Tryolabs   (2) IIE–FING, UDELAR, Uruguay   (3) AIT Austrian Institute of Technology

2nd Deep Learning and Security Workshop

IEEE S&P – May 23, 2019 – San Francisco, CA

# Outline


Learning in NTMA
This Work
Input Represen-tations
Building the datasets
Deep Learning Architectures
Experimental Evaluation & Results
Conclusions

# Learning in NTMA

# Learning in NTMA – why?

- The analysis of network traffic measurements is an **active research field**.
- **Machine learning** models are appealing since we have **tons of data and several problems to solve**.
- Some examples:
  - Traffic prediction and classification
  - Congestion control
  - Anomaly detection
  - Cybersecurity (e.g., malware detection, impersonation attacks)
  - QoE estimation
  - . . .

# Learning in NTMA – which kind of models?

- Traditional –shallow– machine learning **models** are commonly used.

# Learning in NTMA – which kind of models?

- **Traditional –shallow– machine learning models** are commonly used.
- Decision trees and random forest, SVM, k-NN, DBSCAN... the list is **as vast as the associated literature.**

# Learning in NTMA – which kind of models?

- **Traditional –shallow– machine learning models** are commonly used.
- Decision trees and random forest, SVM, k-NN, DBSCAN... the list is **as vast as the associated literature.**
- **Feature engineering needed!**

# Learning in NTMA – which kind of models?

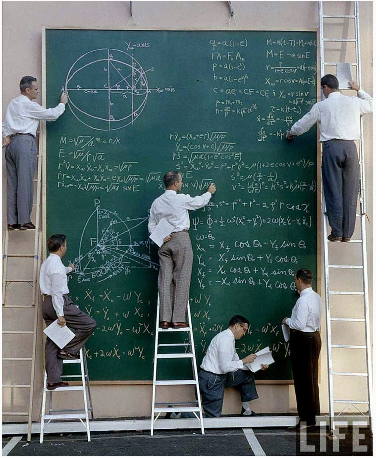
- **Traditional –shallow– machine learning models** are commonly used.
- Decision trees and random forest, SVM, k-NN, DBSCAN… the list is **as vast as the associated literature.**
- **Feature engineering needed!**
- **Handcrafted-expert domain features** are critical to the success of the applied models.

# Learning in NTMA – Expert knowledge

- Features are **heavily dependant on the expert background and the specific problem.**

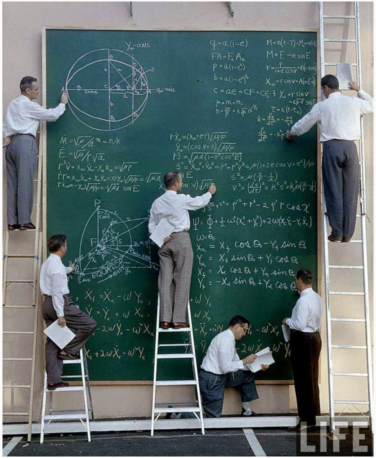
# Learning in NTMA – Expert knowledge

- Features are **heavily dependant on the expert background and the specific problem.**
- Each paper in the literature defines its **own set of input features for the considered problem**, hindering generalization and benchmarking of different approaches.

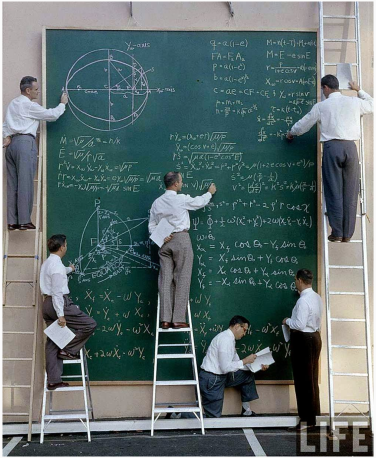


# Learning in NTMA – Expert knowledge

- Features are **heavily dependant on the expert background and the specific problem.**
- Each paper in the literature defines its **own set of input features for the considered problem**, hindering generalization and benchmarking of different approaches.
- Feature engineering is **costly.**



- Features are **heavily dependant on the expert background and the specific problem.**
- Each paper in the literature defines its **own set of input features for the considered problem**, hindering generalization and benchmarking of different approaches.
- Feature engineering is **costly.**
- All in all, **good results can be achieved.**



*Can Deep Learning enhance the presented limitations of traditional models?*

# This Work

# This Work

- **Main goal:** to **explore the end-to-end application** of deep learning models to **complement traditional approaches** for NTMA, using different representations of the input data.

# This Work

- **Main goal:** to **explore the end-to-end application** of deep learning models to **complement traditional approaches** for NTMA, using different representations of the input data.
- To do this, **malware traffic detection and classification** problem is addressed, using *raw*, bytestream-based data as input.

# This Work

- **Main goal:** to **explore the end-to-end application** of deep learning models to **complement traditional approaches** for NTMA, using different representations of the input data.
- To do this, **malware traffic detection and classification** problem is addressed, using *raw*, bytestream-based data as input.
- Research questions

# This Work

- **Main goal:** to **explore the end-to-end application** of deep learning models to **complement traditional approaches** for NTMA, using different representations of the input data.
- To do this, **malware traffic detection and classification** problem is addressed, using *raw*, bytestream-based data as input.
- Research questions
  1. Is it possible to achieve **high detection accuracy** with **low false alarm rates** using the **raw-input, deep learning-based models?**

# This Work

- **Main goal:** to **explore the end-to-end application** of deep learning models to **complement traditional approaches** for NTMA, using different representations of the input data.
- To do this, **malware traffic detection and classification** problem is addressed, using *raw*, bytestream-based data as input.
- Research questions
  1. Is it possible to achieve **high detection accuracy** with **low false alarm rates** using the **raw-input, deep learning-based models?**
  2. Are the proposed models ***better*** than the commonly used shallow models, when **feeding them all with raw inputs?**

# This Work

- **Main goal:** to **explore the end-to-end application** of deep learning models to **complement traditional approaches** for NTMA, using different representations of the input data.
- To do this, **malware traffic detection and classification** problem is addressed, using *raw*, bytestream-based data as input.
- Research questions
  1. Is it possible to achieve **high detection accuracy** with **low false alarm rates** using the **raw-input, deep learning-based models?**
  2. Are the proposed models ***better*** than the commonly used shallow models, when **feeding them all with raw inputs?**
  3. How good are these models **as compared to traditional approaches**, where domain expert knowledge is used to build the set of features?

# Input Representations

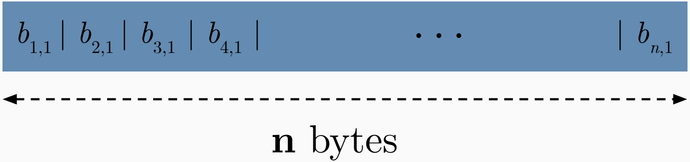
# Raw Packets


- **Decimal normalized representation of each *byte*** of each packet is considered as a different *feature*.

# Raw Packets

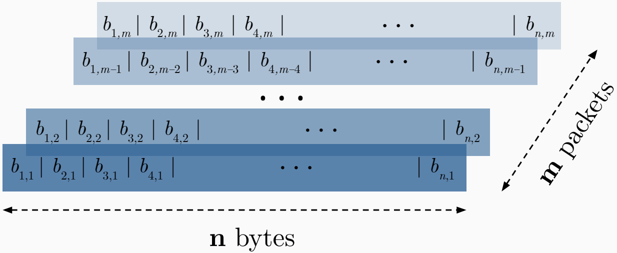

- **Decimal normalized representation of each *byte*** of each packet is considered as a different *feature*.
- **Each *packet*** is considered as a different *instance*.

# Raw Packets



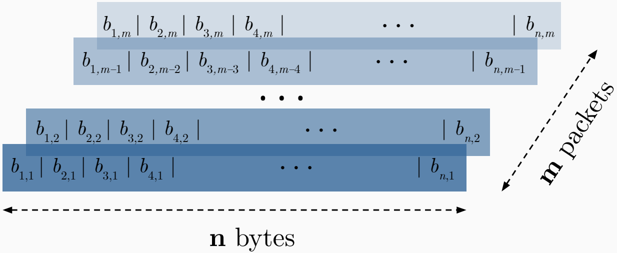

- **Decimal normalized representation of each *byte*** of each packet is considered as a different *feature*.
- **Each *packet*** is considered as a different *instance*.
- It is necessary to choose the **number of bytes** from the packet to be considered (*n*).

# Raw Flows



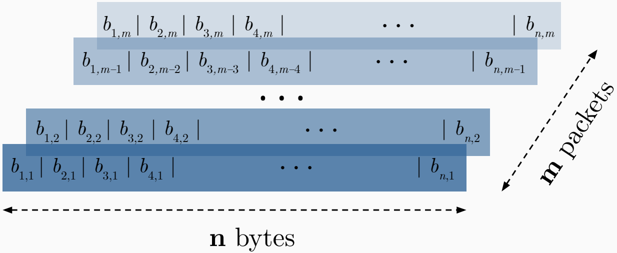

- A *group of bytes* is considered as a different *feature*.

# Raw Flows




- A ***group of bytes*** is considered as a different *feature*.
- **Each *flow*** (group of packets) is considered as a different *instance*.

# Raw Flows




- A *group of bytes* is considered as a different *feature*.
- **Each *flow*** (group of packets) is considered as a different *instance*.
- It is necessary to choose the **number of bytes** from the packet to consider (*n*) and the **number of packets per flow** to consider (*m*).

# Building the Datasets

# Building the Datasets

- **Malware and normal captures** performed by the *Stratosphere IPS Project* of the CTU University of Prague in Czech Republic were considered.

# Building the Datasets

- **Malware and normal captures** performed by the *Stratosphere IPS Project* of the CTU University of Prague in Czech Republic were considered.
- Captures are gathered under **controlled conditions:** fixed scenario (IPs, ports, etc.)

# Building the Datasets

- **Malware and normal captures** performed by the *Stratosphere IPS Project* of the CTU University of Prague in Czech Republic were considered.
- Captures are gathered under **controlled conditions:** fixed scenario (IPs, ports, etc.)
- **Not *in the wild*** network traffic.

# Building the Datasets

- **Malware and normal captures** performed by the *Stratosphere IPS Project* of the CTU University of Prague in Czech Republic were considered.
- Captures are gathered under **controlled conditions:** fixed scenario (IPs, ports, etc.)
- **Not *in the wild*** network traffic.
- Let's consider the **payload**, as the key information to analyze and to build the datasets.

| Representation | Dataset size | n (bytes) | m (packets) |
|---|---|---|---|
| *Raw Packets* | 248,850 | 1024 | N/A |
| *Raw Flows* | 67,494 | 100 | 2 |

**Table 1:** Parameters selection for building the input representation for training the deep learning models.

# Deep Learning Architectures

*Finding the* ***right architecture.***

# Core layers

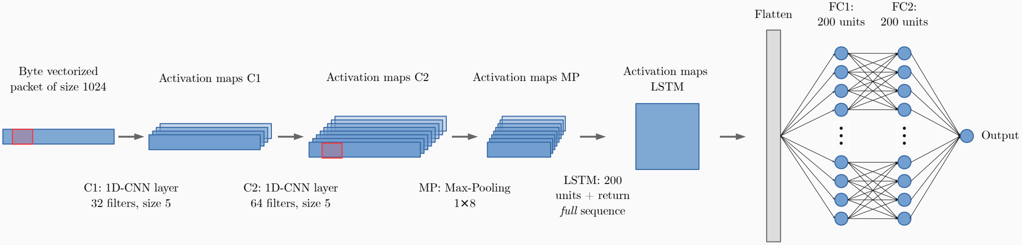
- The core layers used for both models are basically two: convolutional and recurrent.
- **Convolutional**, to **build the feature representation** of the spatial data inside the packets and flows.
- The **recurrent** layers will be used together with the convolutional to improve the performance of *Raw Packets* architecture, allowing the model to keep track of temporal information.
- **Fully-connected** layers to deal with the different combinations of the features in order to arrive to the final decisions (i.e., classify).

# Helper layers

Goals: reduce the generalization error and improve the learning process.

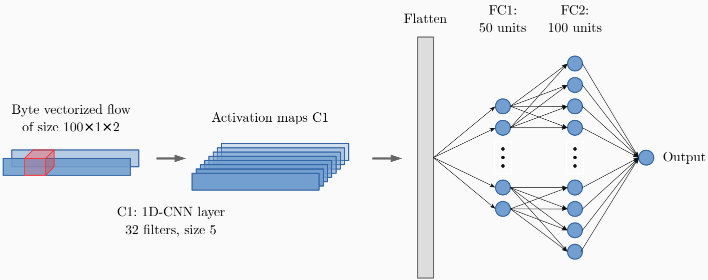
- **Batch Normalization:** layer inputs are normalized for each mini-batch. As a result: higher learning rates can be used, model less sensitive to initialization and also adds regularization.
- **Dropout:** randomly drop units (along with their connections) from the neural network during training. *A very efficient way to perform model averaging:* similar to train a huge number of different networks and average the results.

# *Raw Packets* Architecture




- **2 1D-CNN** layers of 32 and 64 filters of size 5, respectively.
- A **max-pooling** layer of size 8.
- A **LSTM** layer of 200 units, returning the outputs of each cell.
- **2 fully-connected** layers of 200 units each.
- Binary **cross-entropy** is used as the loss function.

# *Raw Flows* Architecture




- **Smaller capacity** than *Raw Packets* (less number of features).
- **1 1D-CNN** layer of 32 filters of size 5 and **2 fully-connected** layers of 50 and 100 units each.
- Also, **binary cross-entropy** is used as the loss function.

# Experimental Evaluation & Results

# *Malware Detection*

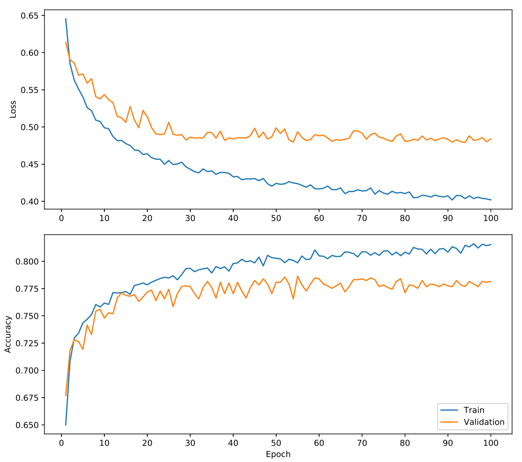
*A First Approach Using Deep Learning*

# Malware Detection: Raw Packets

Detect **malware at packet level**.

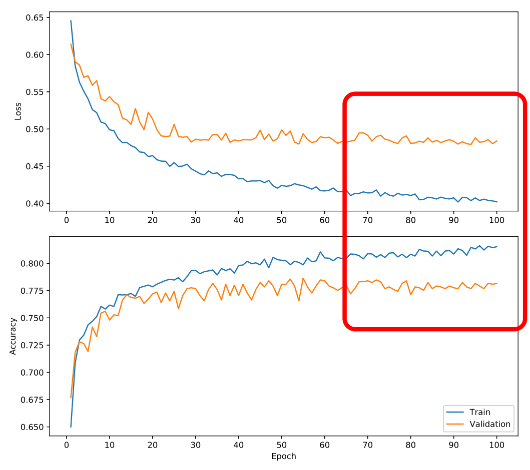
- *Raw Packets* deep learning architecture trained using the respective dataset version ($\sim 250,000$ samples).
- Split using a 80/10/10 schema: 80% for training, 10% for validation and 10% for testing.
- Training held over 100 epochs.
- Adam used as the optimizer function, annealing the learning rate over time.

# Malware Detection: Raw Packets




Raw Packets learning process.

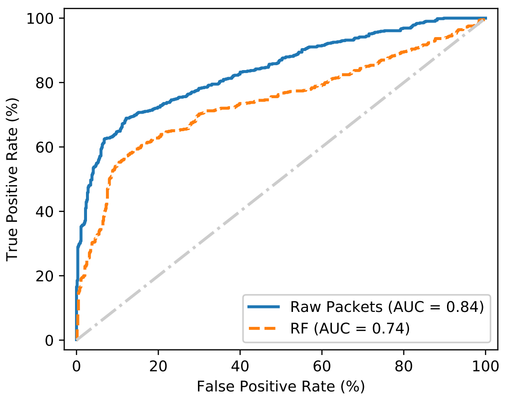
# Malware Detection: Raw Packets




Raw Packets learning process.

# Malware Detection: Raw Packets

- Accuracy: 77.6% **over the test.**
- Comparison with a random forest model (100 trees), using **exactly the same input features**.
- *Raw Packets* deep learning model **outperforms the random forest one.**

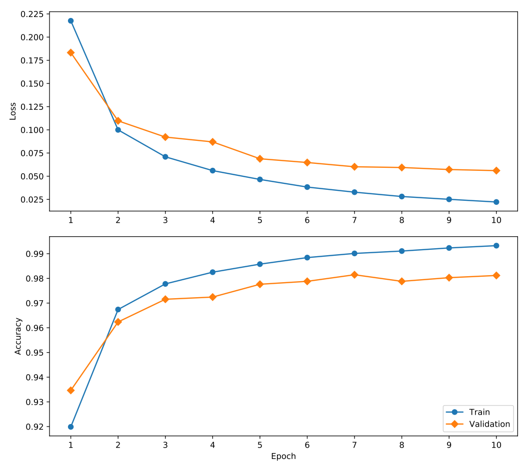


# Malware Detection: Raw Flows

Detect **malware at flow level**.

- *Raw Flows* deep learning architecture trained using the respective dataset version ($\sim 68,000$ samples).
- Split using a 80/10/10 schema: 80% for training, 10% for validation and 10% for testing.
- Training held over 10 epochs.
- Adam used as the optimizer function.

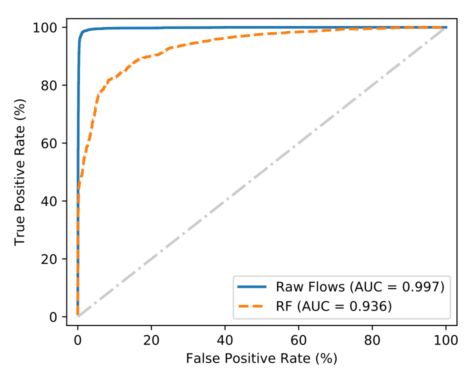
# Malware Detection: Raw Flows




Raw Flows learning process.

# Malware Detection: Raw Flows

- Accuracy: 98.6% **over the test.**
- Comparison with random forest: data was flattened in order to fit the input.
- *Raw Flows* model can detect as much as **98% of all malware flows with a FPR as low as 0.2%.**
- This suggests that operating at flow level, *Raw Flows* can actually provide highly accurate results, applicable in practice.



## Domain knowledge vs. raw inputs

- How good is *Raw Flows* as compared to a random forest trained with a dataset made of expert-handcrafted features?

## Domain knowledge vs. raw inputs

- How good is *Raw Flows* as compared to a random forest trained with a dataset made of expert-handcrafted features?
- **Flow-level features**, such as: traffic throughput, packet sizes, inter-arrival times, frequency of IP addresses and ports, transport protocols and share of specific flags (e.g., SYN packets).

## Domain knowledge vs. raw inputs

- How good is *Raw Flows* as compared to a random forest trained with a dataset made of expert-handcrafted features?
- **Flow-level features**, such as: traffic throughput, packet sizes, inter-arrival times, frequency of IP addresses and ports, transport protocols and share of specific flags (e.g., SYN packets).
- $\sim$ 200 of these features were built to feed a random forest model.

# Domain knowledge vs. raw inputs

- The random forest model using expert domain features achieves highly accurate detection performance: $\sim 97\%$ with FPR less than 1%.

# Domain knowledge vs. raw inputs

- The random forest model using expert domain features achieves highly accurate detection performance: $\sim$ 97% with FPR less than 1%.
- The deep learning-based model using the *Raw Flows* still outperforms this domain expert knowledge based detector.

# Domain knowledge vs. raw inputs

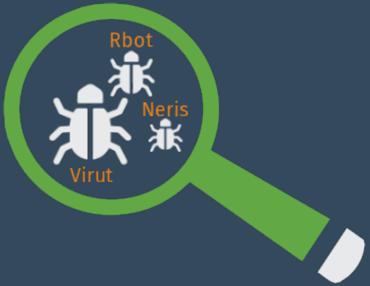
- The random forest model using expert domain features achieves highly accurate detection performance: $\sim$ 97% with FPR less than 1%.
- The deep learning-based model using the *Raw Flows* still outperforms this domain expert knowledge based detector.
- The deep learning model can perform as good as a more traditional shallow-model based detector for detection of malware flows, without requiring any sort of expert handcrafted inputs.

# Other contributions

*From Malware Detection*

*to* ***Malware Classification***

*Please, refer to the paper for further details.*



# Conclusions

# Conclusions

- This work explores **the power of deep learning models to the analysis of network traffic measurements.**

# Conclusions

- This work explores **the power of deep learning models to the analysis of network traffic measurements.**
- The specific problem of **malware network traffic detection and classification is addressed using *raw* representations** of the input network data.

# Conclusions

- This work explores **the power of deep learning models to the analysis of network traffic measurements.**
- The specific problem of **malware network traffic detection and classification is addressed using *raw* representations** of the input network data.
- Using *Raw Flows* as input for the deep learning models **achieves better results** than using *Raw Packets*.

# Conclusions

- This work explores **the power of deep learning models to the analysis of network traffic measurements.**
- The specific problem of **malware network traffic detection and classification is addressed using *raw* representations** of the input network data.
- Using *Raw Flows* as input for the deep learning models **achieves better results** than using *Raw Packets*.
- **In all studied cases**, the deep learning models **outperform a strong random forest model**, using exactly the same input features.

# Conclusions

- This work explores **the power of deep learning models to the analysis of network traffic measurements.**
- The specific problem of **malware network traffic detection and classification is addressed using *raw* representations** of the input network data.
- Using *Raw Flows* as input for the deep learning models **achieves better results** than using *Raw Packets*.
- **In all studied cases**, the deep learning models **outperform a strong random forest model**, using exactly the same input features.
- The *Raw Flows* architecture **slightly outperforms a random forest model trained using expert domain knowledge features.**

THANKS for your attention!

Questions?

@stillyawning

gonzalo@tryolabs.com