Physical Layer Intrusion Detection for Fieldbuses

Andreas Zdziarstek*, Johann Bauer†, Thomas Mundt‡, and Darshit Pandya§
University of Rostock, Germany
Email: {*andreas.zdziarstek, †johann.bauer, ‡thomas.mundt, §darshit.pandya}@uni-rostock.de

***Abstract*—There are multiple widely-used field bus systems for smart buildings with notable security deficiencies versus attackers with physical access to the structure. As there are significant hurdles and costs associated with refitting those systems using cryptographic security, we propose a method of protection through physical-layer intrusion detection. Our implementation is able to identify previously unknown rogue devices through their electrical signatures. To achieve that we implement different models based on autoencoders, a machine-learning concept which can be used for anomaly detection.**

***Index Terms*—Intrusion detection, physical layer security.**

I. Introduction

Fieldbuses connect sensors and actuators with controllers in many areas of automation, e.g. in industrial and building automation, aka. operational technology [1]. Unfortunately, the systems currently in use have several security issues [2] and have been shown to be easily compromised due to a lack of cryptographic safeguards [3]. Most protocols used at fieldbus level are ancient. Update paths towards more secure protocols are often impossible. The systems were thought of as being insular and unconnected to a wider internet, thereby secure. Today, gaining remote access to an unsecured system can be as easy as popping off a light switch in an unseen place and plugging in an internet-connected device with a compatible bus interface.

Prior research has shown promising results in the development of intrusion detection systems [4] for fieldbuses. [5] suggests the physical layer (PHY) of fieldbuses holds promise for uniquely identifying devices by the electrical characteristics of their communication signals, not unlike a voiceprint identification. The PHY intrusion detection system (IDS) would need to be trained for the physical layer characteristics of any number of known devices in the network and raises an alarm if any bus signals are sufficiently different from the known set of devices. What makes this challenging is the fact that we do not know what "different" exactly means in this context, i.e. we have no universal metric to establish a distance between two physical-layer measurements. The only assumptions we can safely make is that the operators know which devices are supposed to be in the network and that the IDS has sufficient opportunity to "learn their voice."

Given all that, machine learning techniques which can operate on mostly raw measurements, learn a model for a given device and then classify something as an in- or outlier are a reasonable choice. Specifically, autoencoders in different variants have recently been used for anomaly detection. The general idea is a multi-layered neural network which is trained to reconstruct its own input at its output. At first glance this sounds a bit counter-intuitive but the application as an anomaly detector arises from the fact that an autoencoder is able to reconstruct those inputs which are similar in some way to the training data better than others. It is also worth mentioning that the training process only uses data labeled as normal. This makes it very suitable for IDS applications as not all classes of anomalies may be known in advance.

II. Related Work

In the area of network anomaly detection, some research has been done on the use of autoencoders, but based on derived upper layer features rather than physical layer measurements. In 2018, Chen *et al.* have found conventional and convolutional autoencoder models to outperform earlier methods of detecting anomalies within the NSL-KDD dataset [6][7].

In [8], Farahnakian and Heikkonen present their implementation of autoencoders with multiple hidden layers for attack detection. Their implementation performed similarly well as the one above. The used data set consists of derived features as well. Another interesting use of autoencoders is presented in [9] by Marchi *et al.* In their work, the authors implement an acoustic novelty detector. This means, given a current input, their model has learned to predict the next samples. If those differ significantly from the real new data, an event is detected.

III. Implementation

As indicated in the first section, our final aim is the implementation of a physical-layer IDS that can be transparently added to existing fieldbus installations. For this work, we have developed an experimental measurement interface for the widely-used KNX building automation fieldbus and multiple anomaly detection models using digital signal processing and autoencoders.

A. KNX Technical Background

KNX is a fieldbus system standard used by smart buildings [10]. While the standard includes a number of different transport media, the most common one is KNX over twisted-pair wiring or KNX-TP [11]. The physical layer defines a resting potential of about 30 V which is used to power smaller bus devices. This resting potential is interpreted as either an idle bus or a logical one. A sending device may generate a logical zero by pulling the bus voltage lower by more than 6 V for exactly 35 µs. After that a choke coil within the bus power supply will cause the voltage to swing above and then decay back to the resting potential. Overall, this takes $104\,\mu s/bit$

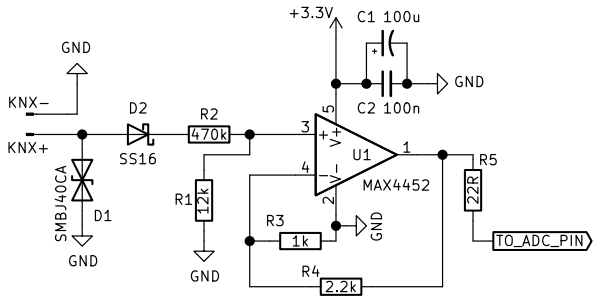


Fig. 1. Schematic diagram of the KNX-to-ADC attenuation and input buffer stage.

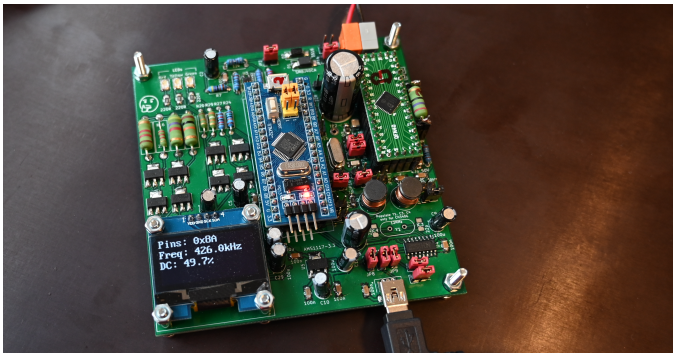


Fig. 2. Prototype device.

or $(9600\,\mathrm{bit/s})^{-1}$. An example measurement can be seen in figure 4a.

B. Hardware

As we intend for our work to be used in add-on devices for existing networks, cost and required effort are factors to be considered here. While it is possible to gather usable physical layer data using a digital oscilloscope [5], cost is prohibitive and the correct use of such equipment is non-trivial. For our setup we decided on working with low-cost microcontrollers and their integrated analog-to-digital converters (ADCs). Specifically, we opted to use the STM32F401CC by ST Microelectronics [12]. Its internal ADC is capable of a $2.4\,\mathrm{MHz}$ sampling rate at a resolution of $12\,\mathrm{bit/sample}$. The complete circuit can be seen in figure 1. Using this input stage, we can achieve very fine-grained measurements over the full voltage range of the KNX signal. The device is depicted in figure 2.

C. Anomaly Detection Models

On the software side, we have implemented four distinct models, three of them based on the autoencoder concept and one, the so-called "Naive Model" based only on correlation between the measurement and a "trained" average. For easy implementation, we used Python with the Keras/Tensorflow libraries.

Originally, we included the Naive Model only as a comparison point not based on machine learning (ML) to verify that

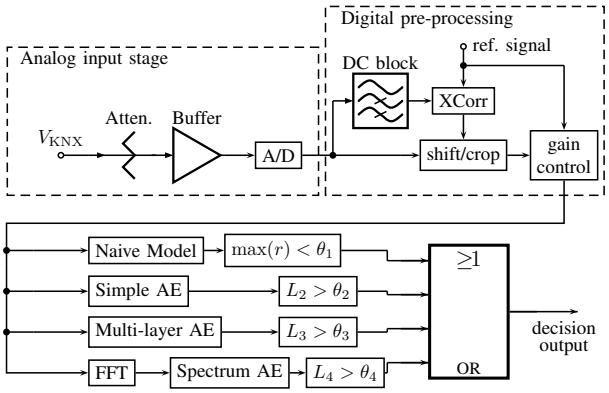


Fig. 3. Signal processing block diagram of the combined model.

ML indeed brings an improvement. Nevertheless, at the end we also included it within a 5th model intended as a sort of combined "super-model" as it yielded a small but measurable gain in accuracy.

All models share a common signal pre-processing part visible in figure 3. Essentially, all new inputs are cross-correlated with a known pre-measured reference signal. The reference is a constant part of an expected response to a serial number request each device gets sent by us. The intention here is to filter out spurious triggers by the ADC and to make sure all measurements are properly overlayed.

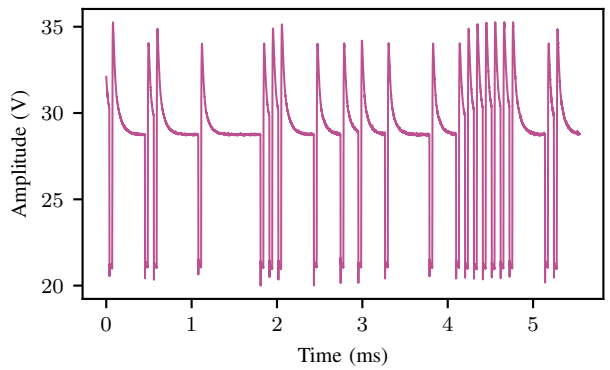
The autoencoder models all return a reconstruction loss L when presented with a sample. During training we set a threshold value θ which is equal to the maximum L encountered for any training sample after the training has concluded. This means there are by design no false positive classifications in the training set and also the false positive rate within the test set should be low to none if everything works as expected. Unless stated we use ReLU as the activation function for the hidden layer and the sigmoid function for the output.

1) Naive Model: Test data is then cross-correlated against it and the maximum correlation $\max(r)$ is the output akin to the reconstruction loss of the autoencoders, just that higher means *more* similar in this case. Accordingly, a test sample needs to score lower than the set threshold θ_1 to be classified as an anomaly.

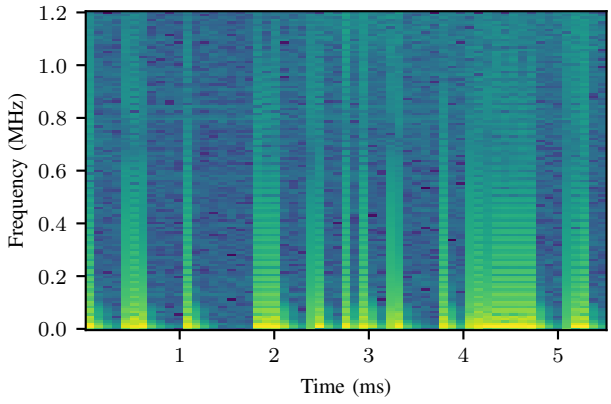
2) Simple Autoencoder: Our first and simplest autoencoder-based model consists of just 3 densely-connected neural network layers. The input and output layers have the same size as the input which in our measurements is always 13312 samples. The inner hidden layer has size 128, sizably reducing the dimensionality.

3) Multi-layer Autoencoder: The multi-layer autoencoder implementation consists of the in- and output layer as above but now with 5 hidden layers. All layers are again densely connected. The sizes of the hidden layers are $\{128, 64, 32, 64, 128\}$ from in- to output.

4) Spectrum Autoencoder: We also implemented an autoencoder model that would normally be used as an image anomaly detector to feed it spectrograms of the measurements,



(a) Measurement plotted as amplitude over time.



(b) Measurement plotted as a spectrogram.

Fig. 4. An example measurement from device 1.2.1.

similar to the acoustic novelty detection in [9]. The spectrogram is likewise generated by performing many short Fourier transforms on the input. Furthermore we applied logarithmic scaling and a normalization to the interval $[0, 1]$ before feeding the resulting image into the autoencoder. The dimensions of the spectrograms given a 13312-sample input at $2.4\,\mathrm{MHz}$ are $59 \times 129 \times 1$ padded to $64 \times 144 \times 1$. An example of such a spectrogram input can be seen in figure 4.

5) Combined Model: During the experimental tests of our models (see next section) it became clear that while there were clear differences in performance between the models, there were certain devices where an otherwise weaker model was outperforming all others. As we also set the anomaly detection threshold generously enough that false positives did occur only very infrequently, this gave us an opportunity to easily combine all the models into a better performing whole. As shown in figure 3, the Combined Model simply contains all other models in parallel running their binary outputs into a logical-OR. So, prosaically this means a signal is classified as an anomaly if and only if at least one of the other models classifies it as one.

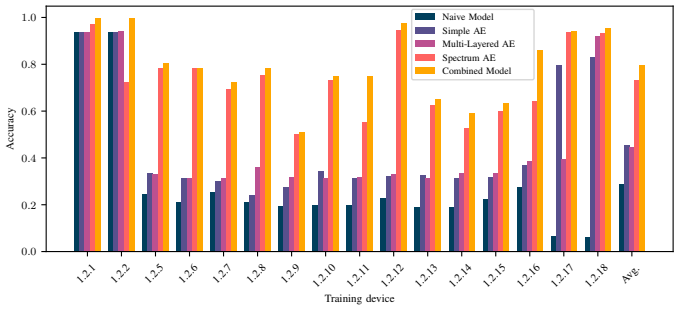


Fig. 5. The accuracy of all tested models for each device and on average.

IV. Experimental Results

The finished setup has been experimentally tested on real-world data gathered by us from a single-room building automation installation consisting of 16 devices. The room and the installed equipment were and are currently in productive use. We used our hardware to gather 480 measurements of the same identification response telegram from each device, resulting in 7680 measurements total. Each one consists of 13312 samples or $5.55\,\mathrm{ms}$ of the bus voltage signal.

For evaluation purposes we trained a classifier for each device and model. In every case we classified the training device as the only known and non-anomalous device, all other samples were labeled as anomalies and only presented as test data. To build a classifier for the whole network from this, a logical-AND between the outputs of the trained classifiers for the known "good" devices would suffice.

We split the data for each device in 120 test and 360 training samples using the same randomization seed each time, to make sure the test samples of one model are not used as training samples for another which could potentially introduce a bias. As a result each model gets 360 "good" measurements of its own device for training and 1920 pieces test data from all devices. Ideally the classifier should then detect 1800 anomalies and 120 known signals.

Each single autoencoder model was trained for 100 epochs with a batch size of 40 using Adaptive Moment Estimation (Adam) for optimization and mean squared error as the loss function.

The plot in figure 5 shows the prediction accuracy for each model and training device, as well as an overall average. Note that the naming stems from KNX hierarchical addresses the devices are using for communication. Though, in this case the given addresses have been altered from their real addresses for data protection reasons as the installation is in active use. As mentioned before, the rate of false positives, meaning a known signal classified as an anomaly, is consistently very low even in the more aggressive Combined Model. In fact, the highest false positive rate encountered in all classifiers was 5 of 120 in the Combined Model for device 1.2.5. On average, the Combined Model classified 2.0625 false positives of 120 negative test measurements. All other models scored better in

TABLE I
THE AVERAGE ACCURACY OF THE TESTED MODELS

N. Mdl.	Sim. AE	M.-L. AE	Spect. AE	Comb. Mdl.
0.289	0.455	0.448	0.732	0.795

this regard—which is expected—because the Combined Model is the worst-case for false positives caused by the logical-OR.

What this means for the results plot is that the accuracy here is in every case dominated by and approximately equal to the true negative rate. In more prosaic terms, the higher the accuracy, the more "bad" signals are detected as such.

There are a few key points to take away from the results in figure 4. Firstly, autoencoders clearly and consistently have an edge versus the trivial correlation approach. Secondly, while the Spectrum Autoencoder is a clear winner versus the other single models overall, there are special cases with some devices where it does not seem so clear. As we took a closer look inside the data, we realized that the results of the different models were often complementary. There are situations where one model clearly performs best on average but another one is actually better at detecting specific devices as anomalies. This is where the Combined Model comes into play. As we can see in table I, the increase in accuracy versus the Spectrum Autoencoder by itself is sizable. In summary, both the Spectrum Autoencoder and of course the Combined Model deliver a very usable performance that could well be leveraged for physical layer intrusion detection.

V. Conclusion and future work.

In this work we introduced the use of autoencoder-based machine-learning models for physical layer security. Furthermore, we subjected our models to comprehensive experimental tests using real-world data gathered using hardware which could conceivably be deployed "in the field" without much associated cost. The results validate our assumption that autoencoders operating on physical layer data can be a significant part of a capable IDS system for fieldbuses. That said, the results also show that the detection rate is good but by no means perfect. To improve on that, it seems sensible to combine our physical layer approach with IDS models operating on higher OSI-layers. Given our final aim of developing a kind of "plug-and-play" IDS, it would also make sense to look into the miniaturization and simplification of the software side. It is conceivable that given enough tuning and compression, the models could be sufficiently reduced to be feasibly deployable on lightweight computing devices.

References

[1] J. King and C. Perry, "Smart Buildings: Using Smart Technology to Save Energy in Existing Buildings," American Council for an Energy-Efficient Economy (ACEEE), Research Report A1701, Feb. 27, 2017. [Online]. Available: https://www.aceee.org/research-report/a1701 (visited on 05/26/2021).

[2] W. Granzer, F. Praus, and W. Kastner, "Security in Building Automation Systems," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 11, pp. 3622–3630, Nov. 2010, ISSN: 1557-9948. DOI: 10.1109/TIE.2009.2036033.

[3] T. Mundt, F. Krüger, and T. Wollenberg, "Who Refuses to Wash Hands? Privacy Issues in Modern House Installation Networks," in *2012 Seventh International Conference on Broadband, Wireless Computing, Communication and Applications*, Nov. 2012, pp. 271–277. DOI: 10.1109/BWCCA.2012.51.

[4] Z. Pan, S. Hariri, and J. Pacheco, "Context aware intrusion detection for building automation systems," *Computers & Security*, vol. 85, pp. 181–201, 2019, ISSN: 0167-4048. DOI: 10.1016/j.cose.2019.04.011.

[5] A. Zdziarstek, W. Brekenfelder, and F. Eibisch, "Using the physical layer to detect attacks on building automation networks," in *Security and Privacy in Communication Networks*, N. Park, K. Sun, S. Foresti, K. Butler, and N. Saxena, Eds., Cham: Springer International Publishing, 2020, pp. 372–390. DOI: 10.1007/978-3-030-63095-9_24.

[6] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Autoencoder-based network anomaly detection," in *2018 Wireless Telecommunications Symposium (WTS)*, 2018, pp. 1–5. DOI: 10.1109/WTS.2018.8363930.

[7] L. Dhanabal and S. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 6, pp. 446–452, 2015.

[8] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, 2018, pp. 178–183. DOI: 10.23919/ICACT.2018.8323688.

[9] E. Marchi, F. Vesperini, S. Squartini, and B. Schuller, "Deep recurrent neural network-based autoencoders for acoustic novelty detection," *Computational Intelligence and Neuroscience*, vol. 2017, Jan. 15, 2017, ISSN: 1687-5265. DOI: 10.1155/2017/4694860.

[10] *KNX on track for success again in 2019: Sector Coupling and IoT in focus*, Press Release, Brussels, Belgium: KNX Association cvba, Jan. 28, 2019. [Online]. Available: https://media.knx.org/feed/file/1050 (visited on 05/26/2021).

[11] F. Sokollik, P. Helm, and R. Seela, *KNX für die Gebäudesystemtechnik in Wohn- und Zweckbau*, 6th ed. VDE Verlag GmbH, 2017, ISBN: 978-3-8007-4033-8.

[12] *STM32F401xB STM32F401xC*, DS9716 Rev. 11, Datasheet, STMicroelectronics, Apr. 19, 2019. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32f401cc.pdf (visited on 05/28/2020).