Device-Centric Firmware Malware Detection for Smart Inverters using Deep Transfer Learning

Syed Raqueed Bin Alvee
Dept. of Electrical Engineering and Computer Science
Texas A&M University-Kingsville
Kingsville, TX, 78363 USA
syed_raqueed_bin.alvee@students.tamuk.edu

BoHyun Ahn
Dept. of Electrical Engineering and Computer Science
Texas A&M University-Kingsville
Kingsville, TX, 78363 USA
bohyun.ahn@students.tamuk.edu

Seerin Ahmad
Dept. of Electrical Engineering and Computer Science
Texas A&M University-Kingsville
Kingsville, TX, 78363 USA
seerin.ahmad@students.tamuk.edu

Kyoung-Tak Kim
Dept. of Electrical Engineering
Soongsil University
Seoul, South Korea
whitechii@soongsil.ac.kr

Taesic Kim*
Dept. of Electrical Engineering and Computer Science
Texas A&M Univeristy-Kingsville
Kingsville, TX, 78363 USA
taesic.kim@tamuk.edu

Jianwu Zeng
Dept. of Electrical and Computer Engineering and Technology
Minnesota State University, Mankato
Mankato, MN, 56001 USA
jianwu.zeng@mnsu.edu

2022 IEEE Design Methodologies Conference (DMC) | 978-1-6654-7999-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/DMC55175.2022.9906538

***Abstract*—Since future power grids are inverter-dominant grids and inverters are getting smarter by incorporating remote access and seamless firmware update, it is anticipated that malware attackers will directly target smart inverters. However, malware threats targeting smart inverters have been less studied yet. This paper explores potential malware attacks targeting smart inverters and proposes a deep transfer-learning (DTL)-based malware detection framework for smart inverters. The proposed DTL method can significantly reduce development time and efforts for an artificial intelligence-based malware detection algorithm while improving detection accuracy. The experimental result shows that the proposed method achieves 98% of firmware malware detection accuracy. This approach will be transformative to other smart grid devices enabling seamless firmware update.**

Keywords—artificial intelligence, cyberattack, cybersecurity, deep transfer learning, firmware security, malware, smart inverter

I. Introduction

With an extremely high penetration of distributed energy resources (DER) in sub-transmission and distribution systems, current power systems are in transition to inverter-dominant power grids [1]. Inverters are getting smarter for additional grid control functions such as fault ride-through, grid voltage support, and reactive power compensation at the grid edge with extensive data exchange and seamless firmware update in cyber-physical environments. However, cyber-attack surface of the smart inverters has been significantly expanding. It is predicted that more sophisticated attackers will directly target smart inverter infrastructures by coordinated malware attacks to manipulate critical power infrastructures [2]. Furthermore, the detection failure of malware injection into smart inverters could spread them into networked power system devices leading to physical, financial, and societal impact. However, patching/managing numerous infected inverters in power grids might be a challenge and significant threat in the future power grids. Therefore, smart inverter cybersecurity and engineering teams need to prepare early combating malware attacks using a device-centric defense approach (i.e., security built-in smart inverter).

This work was supported in part by the Department of Energy (DoE) under award No. DE-EE0009026. Funding was provided in part by U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Solar Energy Technologies Office. The views expressed in the article do not necessarily represent the views of the DoE or the U.S. Government.

Recently, few researchers have investigated firmware security for smart inverters. Maliciously modified controller firmware of a solar inverter could be detected by machine learning (ML) classifiers that are trained and validated by data acquired from custom-built hardware performance counters (HPCs) using assembly firmware codes [3]. However, this method requires an appropriate disassembler. Besides, a blockchain-based controller firmware integrity check method [4] and a controller digital twin method [5] have been proposed. Although these methods will be applied for detecting malicious controller firmware modifications given known integrity and signature, it is hard to detect malware attacks targeting the network device in a smart inverter through authorized access/firmware update such as system malware executed by operating system (OS) and firmware malware executed without OS (e.g., MoonBounce implanted in serial peripheral interface (SPI) flash memory [6]). The firmware malware stored in SPI flash memory could be remotely executed and commanded to download additional malware without the traditional OS dependency. In [7], a smart inverter communicating with online anti-virus scanning servers (e.g., VirusTotal [8]) enables to detect known malware files. However, the inverter requires an interface with the third-party external engines through the internet, and the inverter owners/vendors may pay additional fees for advanced anti-virus services. For example, Kaspersky provides ML methods for an advanced malware detection [9]. The authors have proposed a convolutional neural network (CNN)-based ransomware detection using 2-D grayscale image files converted from binary files, which enables to build a device-level malware file detection without a disassembler [10]. However, existing commercial anti-virus engines and ML-based malware detection models utilizing publicly available malware samples might not be applicable for detecting smart inverter-specified malware types and their obfuscated malware files.

This paper explores potential firmware malware attacks targeting smart inverters and proposes a device-centric proactive firmware malware file detection. Available malware and benign firmware samples are collected based on the target inverter information and through reverse engineering. Binary manipulation methods [11] are adopted to generate firmware malware variants (i.e., obfuscated firmware malware). Then,

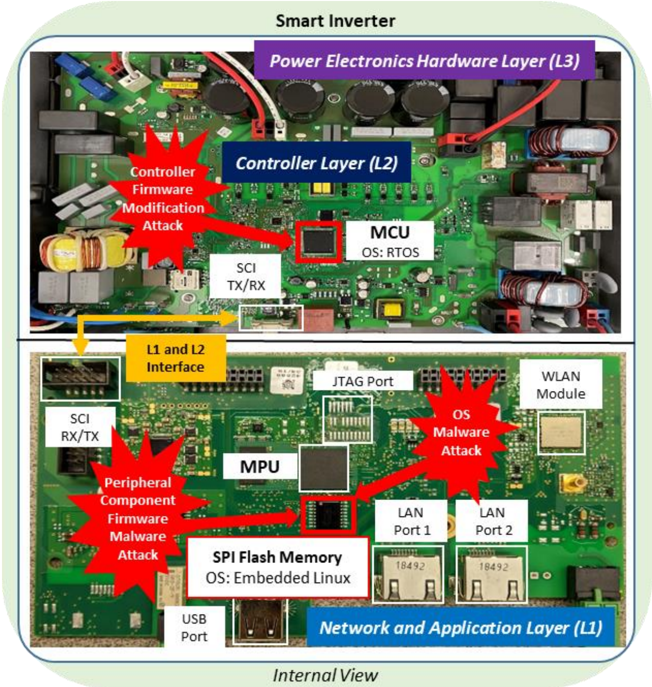


Fig. 1. A commercial smart inverter consisting of three layers: Network and Application Layer (*L1*), Controller Layer (*L2*), and Power Electronics Hardware Layer (*L3*).

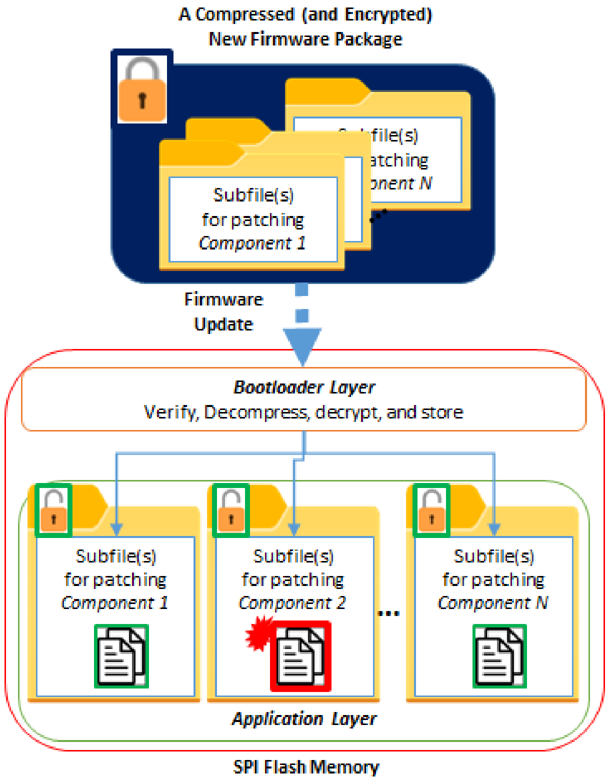


Fig. 2. Firmware update process using SPI flash memory.

the proposed deep transfer learning (DTL) method is applied to design the smart inverter-centric malware detection algorithm by retraining a basis CNN-based malware detection model using the device-specific file samples. The experimental results validate that the proposed method achieves 98% of firmware malware detection accuracy with minimum training efforts and small number of samples, which outperforms the VirusTotal scanning service and the basis CNN model. To the authors' best knowledge, this paper is the first approach for investigating firmware malware in smart inverters and proposing a DTL-based firmware malware detection method.

II. Potential Firmware Malware Attacks and Impacts

A. Smart Inverter System Architecture and Firmware Update Process

Fig. 1 shows a commercial smart inverter that mainly consists of three layers: 1) *Network and Application Layer* (*L1*); 2) *Controller Layer* (*L2*); and 3) *Power Electronics Hardware Layer* (*L3*). *L1* includes a microprocessor unit (MPU) that has higher computational power and operates larger applications with embedded Linux OS and SPI flash memory. Additional peripheral network components such as local area network (LAN) ports, a wireless LAN (WLAN) module, and serial port components are included in this layer. *L2* has a microcontroller unit (MCU) that provides relatively lightweight computing power for signal processing and inverter control with a real-time operating system (RTOS) installed in the on-chip MCU memory. Controller firmware is installed in the MCU. *L3* includes power electronics hardware components and sensors.

The smart inverter firmware versions are generically obtainable from the vendor's official website. Fig. 2 depicts a firmware update process in SPI flash memory of a smart inverter. Depending on the vendor's security policy, the firmware package consisting of multiple types of subfile(s) can be a raw filetype, an encrypted filetype [12], or a compressed filetype [13]. These subfiles include controller firmware, the main OS update or a peripheral component update on the board. A newly received firmware package is automatically verified, decompressed, and decrypted by a secure bootloader [12] installed in SPI flash memory. Afterward, each raw subfile is separately stored in an appropriate application flash section.

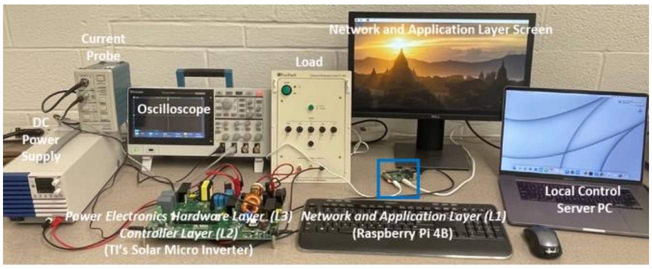
B. Firmware Malware Threat Modeling and Impacts

This paper focuses on firmware malware attacks and their impacts. Firmware malware can be implanted in a device's memory and executed. Table I lists several known vulnerabilities to be used for firmware malware attacks targeting smart inverters based on malware risks in an MCU in [12]. For example, the latest MoonBounce attack tampered with unified extensible firmware interface (UEFI) firmware (i.e., a bootkit) in SPI flash memory [14]. This type of firmware malware is executed during the booting process in *L1*, which can evade the device security functions in OS in the smart inverter. Then, it will facilitate the deployment of user-mode malware attack stage [14]. The adversary can finally upload different malware types in SPI flash memory and impact *L1* in the smart inverter. The attacker then will use the compromised *L1* to manipulate the controller firmware in *L2* through the SCI interface.

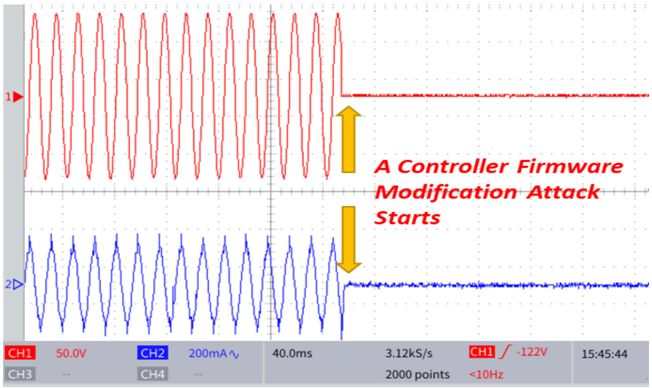
Fig. 3 (a) shows an experimental setup for the penetration testing emulating a smart inverter system environment using an evaluation solar micro inverter and an IoT device. Through the initial access to *L1* using a backdoor malware, the controller firmware is modified to disconnect the relay switch disconnection in the smart inverter. Fig. 3 (b) illustrates the

TABLE I. KNOWN VULNERABILITY OF SMART INVERTER TO BE USED FOR FIRMWARE ATTACKS

Vulnerability	Descriptions	Firmware-related Impacts
1) Fuse and Lock bits	The failure setting of the fuse and lock bits in MPU/MCU/SPI flash memory allows malicious memory reverse engineering via the serial communication ports.	A physical attacker can directly dump firmware images from the flash memory. Furthermore, firmware decryption keys and credentials can be leaked from EEPROM.
2) Clock Glitch	A spike of clock frequency in a short time applied by a maliciously intended high voltage or an extra heat by overclocking to MPU/MCU/SPI flash memory can lead to cryptography function failures.	Firmware malware can be loaded to the flash memory bypassing a cryptography verification in the secure bootloader due to a damaged memory chip.
3) Readback Tool	The purpose of the readback tool is a diagnosis for vendors after dumping firmware out of the chip. Vendors can remotely access the chip through this tool.	Multiple firmware attacks can occur if this tool is maliciously leaked to an adversary or the diagnosis task is implemented under an unsecured internet protocol.



(a)



(b)

Fig. 3. Penetration testing: (a) experiment setup on an emulated smart inverter security testbed and (b) waveforms of the smart inverter.

voltage waveform of the inverter caused by the malware attack. The target inverter stops to generate power.

III. PROPOSED DEEP TRANSFER LEARNING METHOD

This section proposes a DTL-based malware file detection method, specifically improving the detection accuracy for device-centric firmware malware file detection. DTL utilizes a deep learning model designed by malware detection/classification as a basis model. Typically, practitioners would take a pre-trained model from a type of image dataset, freeze a portion of the layers, and fine-tune the last few layers on the newly obtained dataset. The advantages of transfer learning include accelerating training time, reducing parameters and architecture search for deep neural networks, and maintaining high classification performance, especially on the relatively smaller-sized data set such as smart inverter firmware malware. Fig. 3 illustrates the proposed DTL process, which consists of three sequential processes: 1) Selection and training a basis DL model, 2) device-centric data collection and generation, 3) data preprocessing, and 4) processing DTL in the basis DL model. In this paper, VGG 16 model is chosen as a DL basis model, which is trained by publicly available malware files.

Fig. 4 illustrates the proposed DTL process, which consists of three sequential processes: 1) selection and training of a basis DL model, 2) device-centric data collection and generation, 3) data preprocessing, and 4) processing DTL in the basis DL model. In this paper, the Visual Geometry

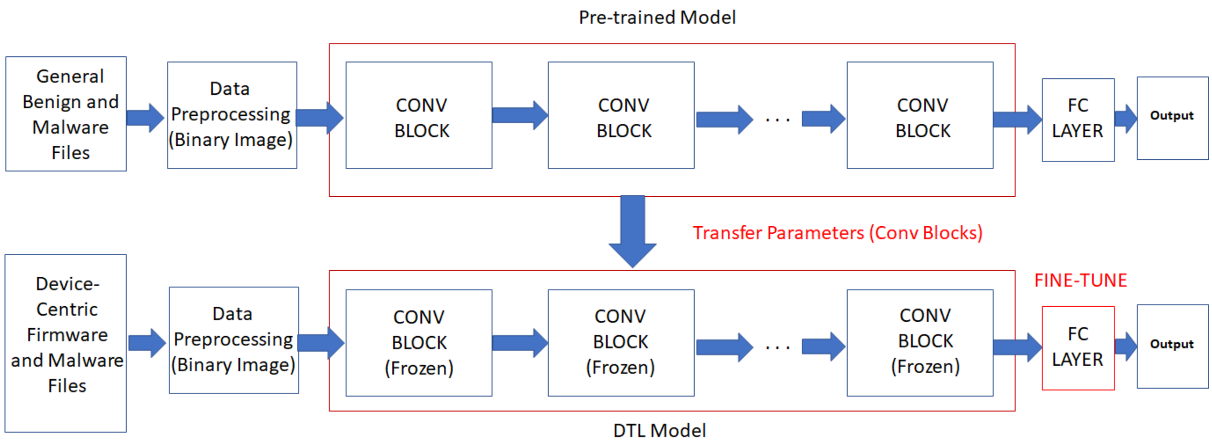


Fig. 4. The proposed Deep Transfer Learning (DTL) model.

Group 16 (VGG 16) model is chosen as a DL basis model, which is pre-trained by publicly available malware files.

A. *Device-centric Data Collection and Generation*

In the embedded system field, a direct firmware extraction technique (i.e., firmware dump or read) from memory is representatively used via 1) UART debugging port, 2) JTAG debugging port, or 3) direct flash memory chip (desoldering if necessary). This paper adopts the direct firmware dump method to extract stored firmware using the vulnerability (1) Fuse and Lock bits) in Table I. After a serial connection between the smart inverter and a reverse engineering PC, target firmware is extracted by exploiting a dump command in *flashrom* open-source tool, as shown in Fig. 5 (a). Fig. 5(b) shows the extracted firmware analyzed by *binwalk* entropy open-source tool. All subfiles are completely decompressed (or decrypted) with displaying values close to 0. After then, binary manipulation methods [11] are used to generate variants of MoonBounce firmware malware.

B. *Data Preprocessing*

Binary files are converted to image files [10]. A static binary firmware file is mapped to an array of integers between 0 and 255. Hence each binary is converted into a one-dimensional array [0; 255]. Then the array is normalized to [0, 1] by dividing by 255. The normalized array is then reshaped into a two-dimensional array. The height of the file is the total length of the one-dimensional array divided by the width. Interest readers may refer to [10] for more detail of the data preprocessing.

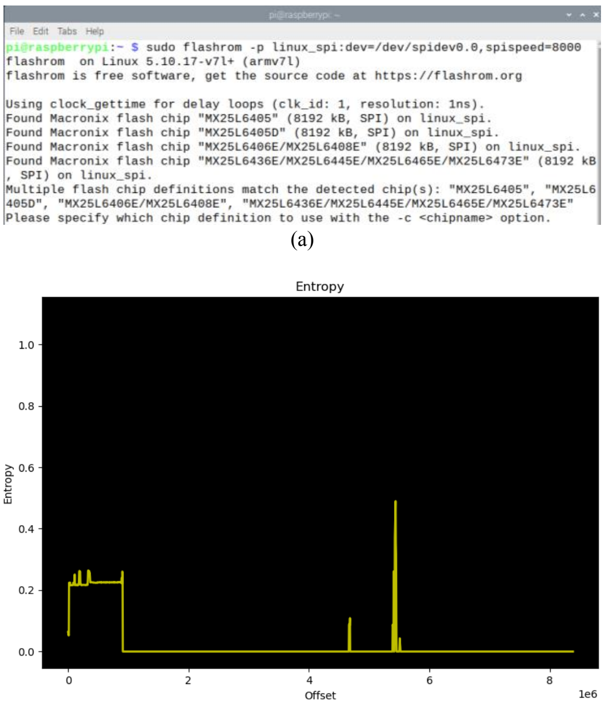


Fig. 5. SPI flash memory dump and analysis test: (a) command to dump binary image subfiles stored in SPI flash memory using *flashrom* tool and (b) checking decrypted binary image subfiles using *binwalk* entropy tool.

C. *DTL using Device-centric Data Samples*

This step is to conduct transfer learning on the reshaped and resized images. The DTL freezes a portion of the DL layers, and the last few layers (not frozen) are retrained on the new image datasets. Fig. 4 shows that the fully connected (FC) layer in the CNN model layers is to find a classification result when a new image dataset is provided. The classification layer is removed and set as not trainable. The hyperparameters of convolution blocks (CONV BLOCK), including loss function, optimizer, and learning rates, are transferred to the DTL model. Also, the preprocess input function is applied to normalize the input data in the model.

IV. Validation

A. *DTL Experimental Setup*

The basis DL model is developed and trained in a cloud computing platform, Jupyter Notebook, using the datasets described in Section III. Firstly, the basis DL model experiment was run on a Windows computer running i7 9750H, RTX 2060, and 16GB RAM. A total of 100 benign firmware files and 100 firmware malware files are used for training (50%) and validation (50%). Then, the proposed DTL model is implemented in an IoT device (64-bit ARM OS, internal GPU (896 MB), and 8GB RAM) using the tested in Fig. 3(a). Finally, the result is compared with VirusTotal [8] and a VGG 16 model (i.e., a DL basis model in this paper) trained by publicly available numerous malware files. The benign and malicious binary files are represented as grayscale images and are used to train the last FC layer in the VGG16 model.

B. *Results*

Fig. 6 illustrates a screenshot of malware file detection results in the VirusTotal server. The anti-virus engines failed to detect a custom-made variant of the MoonBounce malware file. Fig. 7 depicts the accuracy results of the training and validation of the DTL detection model for 40 epochs using the device-centric datasets. Training and validation accuracy values converge to 99% and 98%, respectively. However, the VGG16 basis model shows 91.31% of malware detection accuracy for the device-specified firmware malware samples although the basis malware detection algorithm provides high detection accuracy for the general malware files (> 96%).

Moreover, the CNN model trained by the DTL method is implemented in Raspberry Pi 4B in the testbed. Fig. 8(a) shows the four firmware malware sample variants and one goodware sample and Fig. 8(b) shows that all files are accurately classified. These results show the proposed method is accurate and suitable for device-centric malware file detection for smart inverters.

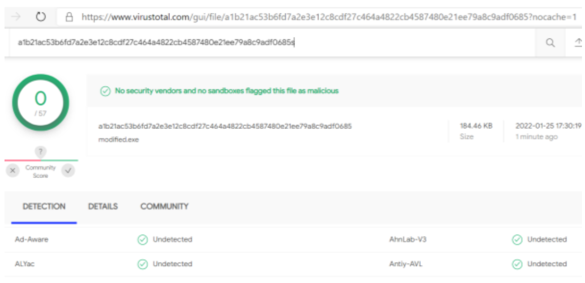


Fig. 6. A screenshot of malware detection results in VirusTotal using a variant sample of MoonBounce.

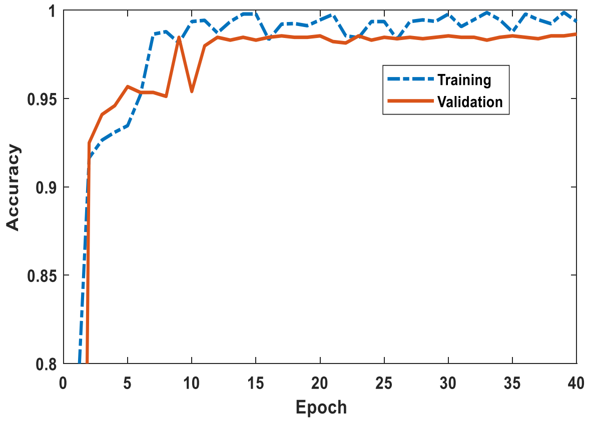
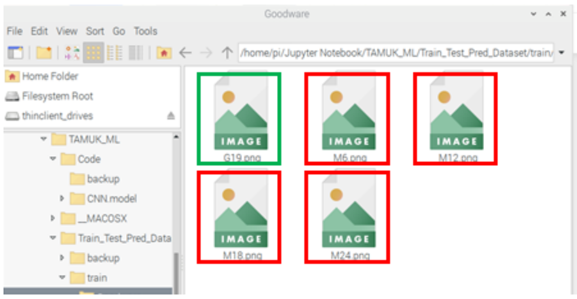
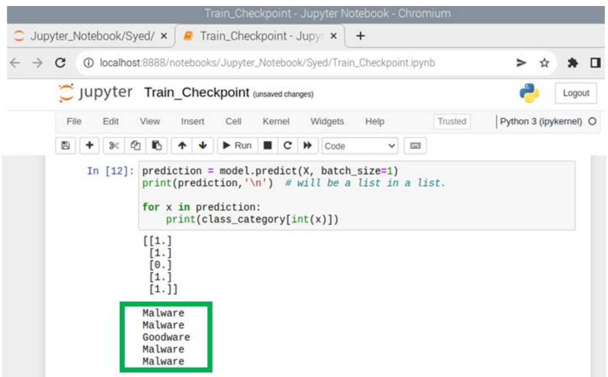


Fig. 7. Training and validation accuracy of the DTL model.



(a)



(b)

Fig. 8. Detection results on an IoT device (Raspberry Pi 4B) using the DTL model: (a) target data (one goodware (benign) and four malware variants) and (b) detection result using Jupyter Notebook.

V. Conclusion

This paper has explored the potential attack surface of firmware malware attacks targeting smart inverters and proposed a deep learning-based malware file detection method using the DTL technique. The proposed method enables to detect firmware malware files with high accuracy with minimum time and effort in a device level. Future works include: 1) investigating more practical attack scenarios, 2) studying on malware variant and obfuscation technique to generate potential firmware malware files, 3) lightweight and high-accuracy detection algorithm development to be implemented in smart inverters, and 4) developing forensic and recovery techniques for the smart inverters.

References

[1] Y. Xue, et al., “On a future for smart inverters with integrated system functions,” in *Proc. 2018 9th IEEE International Symposium on Power Electronics for Distributed Generation systems*, Charlotte, NC, Jun. 25-28, 2018, pp. 1–8.

[2] J. Qi, A. Hahn, X. Lu, J. Wang, and C-C. Liu, “Cybersecurity for distributed energy resource and smart inverters,” *IET Cyber-Phys. Syst., Theory Appl.*, vol. 1, no. 1, pp. 28–39, Dec. 2016.

[3] A. P. Kuruvila, I. Zografopoulos, K. Basu, and C. Konstaninou, “Hardware-assisted detection of firmware attack in inverter-based cyberphysical microgrids,” *International Journal of Electric Power & Energy Systems*, vol. 132, p. 107150, Nov. 2021.

[4] G. Bere, B. Ahn, J. J. Ochoa, T. Kim, A. A. Hadi, and J. Choi, “Blockchain-based firmware security check and recovery for smart inverters,” in *Proc. 2021 IEEE Applied Power Electronics Conference and Exposition (APEC)*, Phoenix, AZ, June 14-17, 2021, pp. 675-679.

[5] S. Moquin, S. Kim, N. Blair, C. Farnell, J. Di, and H. A. Mantooth, “Enhanced uptime and firmware cybersecurity for grid-connected power electronics,” in *Proc. 2019 IEEE CyberPELS Workshop*, Knoxville, TN, April 29-May 1, 2019, pp. 1–6.

[6] M. Lechtic, V. Berdnikov, D. Legezo, and I. Borisov, “MoonBounce: the dark side of UEFI firmware,” SecureList by Kaspersky, Jan. 20, 2022, [Online]. Available: https://securelist.com/moonbounce-the-dark-side-of-uefi-firmware/105468/.

[7] B. Ahn, G. Bere, S. Ahmad, J. Choi, and T. Kim, “Blockchain-enabled security module for transforming conventional inverters toward firmware security-enhanced smart inverters,” in *Proc. 2021 IEEE Energy Conversion Congress and Exposition*, Vancouver, BC, Canada, Oct. 10-14, 2021, pp. 1307-1312.

[8] [Online]. Available: https://www.virustotal.com/gui/home/upload

[9] [Online]. Available: https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf

[10] S. Alvee, B. Ahn, T. Kim, Y. Su, Y-W. Yoon, and M-H. Ryu, “Ransomware attack modeling and artificial intelligence-based ransomware detection for digital substations,” in *Proc. 2021 6th IEEE Workshop on Electronic Grid (eGrid)*, New Orleans, LA, Nov. 8-10, 2021, pp.1-5.

[11] A. Abusnaina, A. Anwar, S. Alshamrani, A. Alabduljabbar, R. Jang, D. Nyang, and D. Mohaisen, “Systemically evaluating the robustness of ML-based IoT malware detectors,” in *Proc. 2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume (DSN-S)*, Taipei, Taiwan, June 21-24, 2021, pp. 3-4.

[12] C. Gao, L. Luo, Y. Zhang, B. Pearson, and X. Fu, “Microcontroller based IoT system firmware security: Case studies,” in *Proc. 2019 IEEE International Conference on Industrial Internet (ICII)*, 2019, pp. 200-209.

[13] S. Lee, J. -Y. Paik, R. Jin, and E. -S. Cho, “Toward machine learning based analyses on compressed firmware,” in *Proc. 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, 2019, pp. 586-591.

[14] S. V. Horn, “Kaspersky uncovers third known firmware bootkit,” Kaspersky, Jan. 20, 2022, [Online]. Available: https://usa.kaspersky.com/about/press-releases/2022_kaspersky-uncovers-third-known-firmware-bootkit.