If you cite this paper, please also cite the main paper: S. Lazzaro, V. De Angelis, A. M. Mandalari and F. Buccafurri, "Is Your Kettle Smarter Than a Hacker? A Scalable Tool for Assessing Replay Attack Vulnerabilities on Consumer IoT Devices", 2024 IEEE International Conference on Pervasive Computing and Communications (PerCom), Biarritz, France, 2024, pp. 114-124, doi: 10.1109/PerCom59722.2024.10494466.

REPLIOT: A Scalable Tool for Assessing Replay Attack Vulnerabilities on Consumer IoT Devices

Sara Lazzaro, Vincenzo De Angelis
Mediterranea University of Reggio Calabria
University of Calabria

Anna Maria Mandalari
University College London

Francesco Buccafurri
Mediterranea University of Reggio Calabria

***Abstract*—REPLIOT is a tool for automatically testing IoT devices vulnerability to replay attacks in the local network. It can be used in the home environment to test a specific IoT device, or in a laboratory to assess replay attack vulnerability on IoT devices on a scale. This artifact shows the usage of REPLIOT in both scenarios.**

I. INTRODUCTION

REPLIOT [1] is an automatic tool for testing replay attacks in a smart home environment. REPLIOT is designed to work with any type of device that communicates with the companion app through the local network. In addition, being our tool device-agnostic, it can also be used in a household environment [2]. To this aim, REPLIOT includes a detection module to automatically detect whether the attack is successful or not.

Through REPLIOT, we performed thousands of automated experiments on 41 commercial IoT devices spanning various vendors and categories. We found that, among these, 15 devices are vulnerable to replay attacks. In addition, through the detection module, we were able to automatically detect whether a device is vulnerable with an accuracy of 0.98-1.

II. REPLIOT

REPLIOT is composed of three modules: (1) a training module, (2) an attack module, and (3) a detection module.

Training Module. This module is aimed at collecting some responses from the target device to train three machine-learning models for novelty detection. We obtain these responses by triggering functions on the device through its companion app and sniffing the traffic.

Attack Module. This module performs a replay attack on a target IoT device. First, the attack module sniffs the traffic exchanged between the legitimate companion app and the target device when a function is triggered on the device. Examples of functions are: switching on/off the light of a smart bulb or watching live from a camera, etc. Then, the collected traffic is organized in *flows* and scheduled in reversed order, as detailed in [1]. Finally, this module replicates the request of each *flow* and stores the related responses.

The two lead authors contributed equally to this work.

This work is partially funded by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU, and the EPSRC PETRAS (EP/S035362/1).

Detection Module. This module is aimed at automatically detecting whether a replay attack is successful or not. This module feeds the first three responses sniffed by the attack module, to a novelty detection model. If all the responses are detected as anomalies then the attack is considered **FAILED**. Otherwise, the attack is considered **SUCCESSFUL**.

III. TOOL REQUIREMENTS

REPLIOT is composed of Python and Bash scripts.

To use the artifact, the following hardware and software requirements must be satisfied:

- An access point (AP) equipped with Linux-based OS. We used a laptop (no special hardware requirements are needed) equipped with Ubuntu 20.04 configured as AP. To use REPLIOT, clone the repository at the link https://github.com/SafeNetIoT/ReplayAttack in a folder on the AP.
- An IoT device and a smartphone in which the companion app of the device is installed. Both the IoT device and the smartphone need to be connected to the same network through the AP. For the attack to be successful the IoT devices should exchange traffic with the companion app within the same network.
- Python version ≥ 3.8
- Tshark version $\geq 4.0.4$
- A set of Python dependencies that can be installed through the following command:

```
pip3 install -r requirements.txt
```

- Android Debug Bridge (adb). It is optional and used just for running REPLIOT in the laboratory.

IV. RUNNING REPLIOT IN THE HOME ENVIRONMENT

We describe how to use REPLIOT in the Home Environment for testing replay attack vulnerabilities of IoT devices. We provide a practical demonstration at https://www.youtube.com/watch?v=TXSQ9XJ8Rpc.

A. Setup

No setup operations are needed.

B. Training Phase

To enable the detection module a training phase is needed. Execute the following steps:

1) Select an **OBVERSE** and the related **REVERSE** state of the device;
2) Move to the folder `REPLIOT/Training/`;
3) Launch the following command on a bash shell:

```
bash Training_device_home.sh
    ↪ MAC_DEVICE INTERFACE
    ↪ MAC_SMARTPHONE SNIFF_TIME
```

   For a detailed description of the input parameters see Section IV-C;
4) When REPLIOT displays in console *"Start triggering the device"*, the user needs to set the device in the **OBVERSE** and **REVERSE** state alternatively through the companion app. This procedure should be repeated at least five times.

C. Attack Module

To launch a replay attack, execute the following steps.

1) Select an **OBVERSE** and the related **REVERSE** state of the device [1] (e.g., the **OBVERSE** state is "device ON", while the **REVERSE** state is "device OFF");
2) Move to the folder `REPLIOT/Test/`;
3) Launch the following command on a bash shell:

```
bash Test_device_home.sh MAC_DEVICE
    ↪ INTERFACE MAC_SMARTPHONE
    ↪ SNIFF_TIME DELAY_TIME DETECTION
```

   where:
   - MAC_DEVICE is the MAC address of the device under test.
   - INTERFACE is the network interface on which the traffic is sniffed.
   - MAC_SMARTPHONE is the MAC address of the smartphone.
   - SNIFF_TIME is the time (seconds) during which the tool sniffs the traffic.
   - DELAY_TIME is the time after which the tool starts the replay attack.
   - DETECTION can be "YES" or "NO". If "NO", REPLIOT does not automatically check whether the attack is successful. If "YES", a preliminary training phase is needed (see Section IV-B).
4) When the tool displays in the console *"Start triggering the device"*, set the device in the **OBVERSE** state (e.g., turn ON the device) through the companion app;
5) When the tool displays in console *"Sniffing completed. The attack will start in DELAY_TIME seconds"*, set the device in the **REVERSE** state through the companion app (e.g., turn OFF the device);
6) If DETECTION is `YES`, the tool reports in console the result of the attack. Otherwise, manually check the device to see whether the attack is successful (e.g., the device is now ON).

V. Running REPLIOT in the Laboratory

We used REPLIOT in the laboratory to assess the vulnerability of 41 IoT devices. Essentially, the main difference between laboratory usage and home usage is that the manual operations performed by the users (i.e., trigger the device through the companion app) are automatically performed through adb. This requires the execution of some preliminary steps to let REPLIOT automatically check whether a command triggered on the smartphone, via adb, is successfully executed. Details are provided at https://github.com/SafeNetIoT/ReplayAttack. After the setup, to execute the *Training Phase*, move to the folder `REPLIOT/Training/` and launch the following command on a bash shell:

```
bash Training_device.sh SERIAL_NUMBER
    ↪ MAC_DEVICE INTERFACE PACKAGE
    ↪ MAC_SMARTPHONE
```

where:

- SERIAL_NUMBER is the serial number of the phone in adb.
- PACKAGE is the package name of the companion app.

The other inputs are the same as Section IV-C.

To execute the *Attack Module*, move to the folder `REPLIOT/Test/` and launch the following command on a bash shell:

```
bash Test_device.sh SERIAL_NUMBER
    ↪ MAC_DEVICE INTERFACE PACKAGE
    ↪ MAC_SMARTPHONE
```

VI. Data and Log

The results of the experiments conducted in [1] are publicly available on GitHub at https://github.com/SafeNetIoT/ReplayAttack/tree/main/REPLIOT/Data. There are 15 folders (one for each device found vulnerable). In each folder, within the subfolder `Experiments/Real_Time`, there are 50 folders corresponding to 50 simulations conducted on that device. Each simulation includes three files: the final result of the attack, a log file of the execution of our tool, and a pcap file containing the traffic sniffed to launch the attack. We additionally tested the vulnerability of IoT devices in the **Restart Scenario** [1]. The results of this scenario are available in `Experiments/Delayed`.

References

[1] S. Lazzaro, V. De Angelis, A. M. Mandalari, and F. Buccafurri, "Is your kettle smarter than a hacker? a scalable tool for assessing replay attack vulnerabilities on consumer iot devices," in *IEEE International Conference on Pervasive Computing and Communications*, 2024.

[2] G. Anselmi, A. M. Mandalari, S. Lazzaro, and V. De Angelis, "Copsec: Compliance-oriented iot security and privacy evaluation framework," in *Proc. of the International Conference on MobiCom*, 2023, pp. 1–3.