ChatGPT

SPI Flash Auto-Unlocker

This repository contains `spi_flash_auto_unlocker.py`, a Python utility for automatically dumping, analysing and modifying laptop and embedded firmware to remove BIOS/UEFI passwords. The tool is designed with a *maximalist* ethos, providing verbose output, extensive comments and a retro command-line interface reminiscent of early Macintosh/Web 1.0 environments.

Motivation

Modern laptops often store administrator and user passwords inside the UEFI firmware on the same SPI flash chip that contains the rest of the BIOS. In many cases these passwords are held within a non-volatile variable named `AMITSESetup` in AMI firmwares [1]. Research also shows that certain vendors, such as Lenovo, control write-protection using special variables like `cE!` inside proprietary GUID namespaces [2]. Clearing or deleting these variables effectively resets the password and disables the lockout mechanisms [3]. Rather than manually dumping the flash, editing the image with a hex editor and reflashing, the Auto-Unlocker automates the entire workflow with a single command.

How It Works

1. **Hardware Connection** – You clip a SOIC-8 test clip onto the system's SPI ROM chip and connect it to a compatible USB programmer (CH341A or FT2232H).
2. **Flash Dump** – The script invokes the external `flashrom` utility to read the entire contents of the SPI flash and writes it to a backup file.
3. **Variable Enumeration** – The dumped image is scanned for UEFI variable headers. The parser searches for known GUIDs (e.g. the AMI `AMITSESetup` GUID) and reconstructs the variable name, attributes and data structure. Knowledge of variable header formats and state flags comes from public documentation [4].
4. **Password Removal** – For each variable that is known to contain a password the tool can either zero out the data region or mark the variable as deleted. Setting the state's delete bit (changing `0x3F` → `0x3D`) follows the UEFI specification's life cycle for variable deletion [4]. Clearing the `AMITSESetup` data removes both the user and administrator password [1]. Clearing Lenovo's `cE!` variable disables the backdoor SPI protections [2].
5. **Reflash** – Finally the patched image is written back to the SPI chip with `flashrom`, restoring a system without the old password. You can also opt to perform a dry run and flash manually later.

Usage

First install `flashrom`, Python 3 and any other dependencies. Then run the tool with appropriate options:

```
python3 spi_flash_auto_unlocker.py \
    --reader ch341a_spi       # or ft2232_spi
```

```
--chip W25Q128FV          # optional chip type
--dump backup.bin         # where to store the flash dump
--patch patched.bin       # where to write the patched file
--delete                  # also mark variables as deleted
--no-flash                # do not automatically reflash
```

Running without `--no-flash` will dump, patch and immediately write the modified firmware back to the chip. Always keep the original dump in case something goes wrong. You can also list all detected variables using `--list`.

> **Warning**: Flashing modified firmware can brick your device. Use this software at your own risk and only on hardware you own.

Files

- **spi_flash_auto_unlocker.py** – The main program that implements the dump/patch/flash pipeline. It contains detailed docstrings, functions for GUID conversion, a simple UEFI variable parser, and patch routines.
- **README.md** – This document.

References

- Description of the AMITSESetup variable storing user and admin passwords within the NVRAM [1].
- Discussion of the Lenovo backdoor variable `cE!` used to disable SPI write protections [2].
- Research showing that clearing the `AMITSESetup` variable removes the administrator password [3].
- Explanation of UEFI variable header state values and the deletion process from Count Chu's blog [4].

License

This project is released into the public domain. Use it freely but heed the warnings about safety and legality.

---

[1] Recovering the BIOS password from a Panasonic CF-U1 mk2 (AMI Aptio UEFI) · GitHub
https://gist.github.com/en4rab/550880c099b5194fbbf3039e3c8ab6fd

[2] When "secure" isn't secure at all: UEFI vulnerabilities in Lenovo laptops
https://www.welivesecurity.com/2022/04/19/when-secure-isnt-secure-uefi-vulnerabilities-lenovo-consumer-laptops/

[3] Delving Deep into Reverse Engineering of UEFI Firmwares via Human Interface Infrastructure
https://www.mdpi.com/2079-9292/12/22/4601

[4] Count Chu: The Life Cycle of a UEFI Variable in Flash Part
https://countchu.blogspot.com/2014/09/the-life-cycle-of-uefi-variable-in.html