Monthly Research 2016.7
プラットフォームセキュリティ評価フレームワーク”CHIPSEC”について

F F R I , Inc.
http://www.ffri.jp

E-Mail: research-feedback[at]ffri.jp
Twitter: @FFRI_Research

目次

CHIPSECの概要

- Intelが開発したハードウェアセキュリティの評価ツール
  - 主にセキュリティ技術者向けのツールである
  - ハードウェアセキュリティの調査が行える
    - サイバー攻撃を受ける恐れのある脆弱な設定の検出
  - PCのBIOS/UEFIの設定、構成が主な検査対象
  - Windows、Linux、UEFI Shellで利用可能
- 検査項目がモジュール化されている
- メインのセキュリティチェック機能以外にもハードウェアに対するユーティリティが充実している
  - ROMダンプや書き込み
  - 接続されているPCI Interfaceの情報の取得
- GPLv2ライセンスの下、GitHub上で開発されている
- Pythonで開発されている

検査項目(一部抜粋)

- SMRAM Locking/SPI Controller Locking/BIOS Interface Locking
  - Locking系はコントローラーの設定がロックされているかの検査
  - ロックされていない場合、設定の変更が可能となり、マルウェアによって悪用される危険性
  - マルウェアによって設定が書き換えられた場合、PCの起動不能などの被害が発生する恐れがある
- BIOS Keyboard Buffer Sanitization
  - キーボードバッファーについての確認
  - バッファーが残っている場合はパスワードの流出に繋がる可能性がある
- SMRR Configuration
  - SMRR(System Management Range Register)に対する保護状態のチェック
  - 設定が変更できる状態だとRootkitがインストールされる危険性

検査項目(一部抜粋)

- BIOS Protection
  - BIOSの設定変更に関するロックの検査
  - マルウェア等によってBIOSの設定が変更されるとPCが起動しなくなる恐れ
- Access Control for Secure Boot Keys
- Access Control for Secure Boot Variables
  - Secure Bootに関係する設定の検査
  - 問題がある場合、Secure Bootがバイパスされるなどの脅威が想定される

インストール手順

1. Pythonのインストール
2. 各種Python用パッケージのインストール
   - pwin32
   - Wconio(検査内容によっては必要)
   - py2exe(検査内容によっては必要)
3. ドライバ署名チェックの無効化
   - bcdedit /set TESTSIGNING ON
   - 再起動
4. CHIPSEC用ドライバへの署名 & インストール
   - sc create chipsec binpath=<PATH_TO_CHIPSEC_SYS> type= kernel DisplayName= "Chipsec driver
   - sc start chipsec

※ 詳細なインストール手順についてはCHIPSECのマニュアルを参照

CHIPSECの使い方(Windows)

- chipsec_main.pyを実行すると検査が開始される
  - 引数に検査したい項目のモジュールを指定すると該当の検査が実行される
    - BIOSのロック検査
      - python chipsec_main.py -m common.bios_wp
    - SPI Memoryのロック検査
      - python chipsec_main.py –m common.spi_lock など
  - 検査が完了するとサマリーが表示され、検査結果の詳細が確認できる
    - 検査結果に問題がない場合は緑文字でPASSEDと表示される

- chipsec_util.py を利用すると各種ハードウェアに関する情報が取得できる
  - SPI Memory Dump
    - python chipsec_util.py spi dump
  - PCIカードのROMダンプ
    - python chipsec_util.py pci dump

検査結果の確認

- 検査結果のサマリーに検査の詳細が出力される
  - 本画像はSPI Lockの検査結果
    - 緑文字でPASSEDと表示されている場合は検査結果に問題なし

```
[x][ ================================================================================
[x][ Module: SPI Flash Controller Configuration Lock
[x][ ================================================================================
[*] HSFS = 0xF00C << Hardware Sequencing Flash Status Register (SPIBAR + 0x4)
    [00] FDONE              = 0 << Flash Cycle Done
    [01] FCERR              = 0 << Flash Cycle Error
    [02] AEL                = 1 << Access Error Log
    [03] BERASE             = 1 << Block/Sector Erase Size
    [05] SCIP               = 0 << SPI cycle in progress
    [13] FDOPSS             = 1 << Flash Descriptor Override Pin-Strap Status
    [14] FDV                = 1 << Flash Descriptor Valid
    [15] FLOCKDN            = 1 << Flash Configuration Lock-Down
[+] PASSED: SPI Flash Controller configuration is locked

[CHIPSEC] ***************************  SUMMARY  ***************************
[CHIPSEC] Time elapsed          0.016
[CHIPSEC] Modules total         1
[CHIPSEC] Modules failed to run 0:
[CHIPSEC] Modules passed        1:
[+] PASSED: chipsec.modules.common.spi_lock
[CHIPSEC] Modules failed        0:
[CHIPSEC] Modules with warnings 0:
[CHIPSEC] Modules skipped 0:
[CHIPSEC] ******************************************************************
```

データの解析 –PCI ROM-

- chipsec_util.pyを利用するとPCI ROMのダンプが行える
  - ダンプ内容から接続されている各PCIデバイスの情報が取得できる
    - 例: 先頭からの2byteはベンダIDを表している(リトルエンディアン) この場合はIntel

```
[pci] PCI device 00:00.00 configuration:
     00  01  02  03  04  05  06  07  08  09  0A  0B  0C  0D  0E  0F
00 | 86  80  00  0C  06  00  90  20  06  00  00  06  00  00  00  00
10 | 00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
20 | 00  00  00  00  00  00  00  00  00  00  00  00  43  10  34  85
30 | 00  00  00  00  E0  00  00  00  00  00  00  00  00  00  00  00
40 | 01  90  D1  FE  00  00  00  00  01  00  D1  FE  00  00  00  00
50 | 11  02  00  00  19  00  00  00  17  00  10  DF  01  00  00  DA
60 | 05  00  00  F8  00  00  00  00  01  80  D1  FE  00  00  00  00
70 | 00  00  00  FF  01  00  00  00  00  0C  00  FF  7F  00  00  00
80 | 10  11  11  00  00  11  11  00  1A  00  00  00  00  00  00  00
90 | 01  00  00  FF  01  00  00  00  01  00  D0  1F  02  00  00  00
A0 | 01  00  00  00  02  00  00  00  01  00  E0  1F  02  00  00  00
B0 | 01  00  20  DB  01  00  00  DB  01  00  00  DA  01  00  20  DF
C0 | 00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
D0 | 00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
E0 | 09  00  0C  01  61  E0  04  62  D0  00  54  44  00  00  00  00
F0 | 00  00  00  00  C8  0F  03  00  00  00  00  00  00  00  00  00
```

データの解析 –CMOS Memory-

- CMOS MemoryにはBIOSの設定内容が記憶されている
  - Memory Mapを読むことによってそれぞれのアドレスに格納されているデータが何を表しているのかを知ることができる
  - 赤枠は設定されている日付と時刻を表している(2016/07/22 10:32:48)

```
[CHIPSEC] Dumping CMOS memory..
Low CMOS memory contents:
     00  01  02  03  04  05  06  07  08  09  0A  0B  0C  0D  0E  0F
00 | 48  13  32  07  10  04  05  22  07  16  26  02  50  80  00  00
10 | 00  FF  FF  FF  FF  7F  02  FF  FF  FF  FF  FF  FF  FF  FF  FF
20 | FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  1B  66
30 | FF  FF  20  FF  FF  36  0C  FF  FF  FF  FF  FF  FF  FF  0B  18
40 | 00  00  C0  17  41  28  F0  00  00  10  01  00  00  00  00  00
50 | 00  25  21  00  25  24  23  25  00  00  00  00  00  00  00  00
60 | 00  00  17  00  00  00  00  F0  00  00  00  00  00  00  00  00
70 | 00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
```

考察

- 脅威に対する検査がモジュール化されていることや、検査を実行するコマンドがシンプルであることなどによって、BIOS/UEFI等の低レイヤーについて熟知していなくても利用できるため比較的扱いやすいといえる

- BIOS/UEFI等などの低レイヤー領域を検査するツールは数が少なく 、本ツールは機能が豊富なのでBIOS/UEFIの検査に有効に利用できる可能性がある
  - 検査モジュールの作成もサポートしているため、独自の検査ツールと統合して利用するなどの利用方法が想定される

- ユーティリティを利用するとROMのダンプや書き込みなど、通常のツールでは行えないような事ができ、セキュリティの研究などに有用なことが行える

参考情報

- CHPSECのGitHubページ
  - https://github.com/chipsec/chipsec
- CMOS Memory Map - BIOS Central
  - http://www.bioscentral.com/misc/cmosmap.htm
- CHIPSEC Platform Security Assessment Framework
  - BlackHat2014
  - https://www.blackhat.com/docs/us-14/materials/arsenal/us-14-Bulygin-CHIPSEC-Slides.pdf
- A Tour of Intel CHIPSEC
  - http://www.basicinputoutput.com/2016/05/a-tour-of-intel-chipsec.html
- Malicious Code Execution in PCI Expansion ROM
  - http://resources.infosecinstitute.com/pci-expansion-rom/