Firmware Tools for Security Researchers

BsidesPDX.org
October 15, 2016
Portland, Oregon

Lee Fisher

Agenda

- Next 2 dozen slides: 1 slide per tool.
- Focus:
  - UEFI-flavored system firmware.
  - Tools for live/online and offline analysis of UEFI system firmware.
  - Most are Linux or Windows tools; a few are UEFI Shell tools.
  - Intel x64 is my main emphasis, but many UEFI tools also work on Intel x86, ARM AArch32, and AArch64.
- Non-focus:
  - Not covering BIOS or U-Boot or coreboot style system firmware.
  - Not covering embedded Linux file system image style 'firmware'.
- Time constrainted: only 20 minutes to talk, so only a handful of tools are listed. There are other good tools not listed here, sorry for omissions.

CHIPSEC

- https://github.com/chipsec/chipsec
- CHIPSEC is a framework for analyzing the security of Intel x86 and x64 systems including hardware, system firmware (BIOS/UEFI), and platform components. CHIPSEC does both online analysis of live systems – bare metal and multiple virtualized targets – as well as offline analysis of system images. It includes a security test harness with multiple security modules. It can be run on Windows, Linux, Mac OS X and UEFI shell. CHIPSEC_main is a set of security tests, roughly one module per public vulnerability. CHIPSEC_util is a collection of tools – including fuzzers – to explore a system, eg. to dump rom.bin via SPI. Main and Util share a common "HAL" driver, a native OS kernel driver, for accessing various low level interfaces, and forensic capabilities. The Python-based tool also includes a library that other tools can use. The CHIPSEC Project is part of Intel, maybe now part of the McAfee spinoff?

FWTS (FirmWare Test Suite)

- https://wiki.ubuntu.com/FirmwareTestSuite/Reference
- Firmware Test Suite (fwts) comprises of over fifty tests that are designed to exercise and test different aspects of a machine's firmware. The tools read UEFI, BIOS, ACPI, and other system. FWTS was created by Canonical to help test systems, and works on Ubuntu, and other Linux systems but not BSD or Windows. FWTS has a Linux kernel driver to test UEFI Runtime Services. FWTS has both a command line and a CURSES UI. FWTS also has a liveboot Linux distribution called FWTS-live which can be used to run the tests, using the CURSES UI. LUV also includes FWTS. LUV also includes and runs FWTS in batch mode.The UEFI Forum, which owns ACPI specs, suggests that OEMs run FWTS's ACPI tests.

UEFITool

- https://github.com/LongSoft/UEFITool
- UEFITool is a powerful cross-platform C++/Qt program for parsing, extracting and modifying UEFI firmware images. It supports parsing of full BIOS images starting with the flash descriptor or any binary files containing UEFI volumes. UEFITool is a Qt GUI tool, but the project also includes a few Qt-free C++ command line tools, UEFIDump, UEFIExtract, and UEFIPatch. The main parsing engine and most of the command line tools are not Qt-dependent. (UEFITools' 'UEFIDump' is like a non-GUI version of UEFITool, and is different from FWTS's 'uefidump'.)
- For an example of using UEFITool, look at Intel Security's Advanced Threat Research team's blog post with analysis of the Hacking Team's UEFI malware, they use CHIPSEC and UEFITool to analyze it.

UEFI Firmware Parser

- https://github.com/theopolis/uefi-firmware-parser
- https://pypi.python.org/pypi/uefi_firmware
- UEFI Firmware Parser -- called "uefi_firmware" on Python.org's Cheese Shop -- is a Python module and set of scripts for parsing, extracting, and recreating UEFI firmware volumes. This includes parsing modules for BIOS, OptionROM, Intel ME and other formats too. It supports: UEFI Firmware Volumes, Capsules, FileSystems, Files, Sections parsing, Intel PCH Flash Descriptors, Intel ME modules parsing (ME, TXE, etc), Dell PFS (HDR) updates parsing, Tiano/EFI, and native LZMA (7z) [de]compression, Complete UEFI Firmware volume object hierarchy display, Firmware descriptor [re]generation using the parsed input volumes, and Firmware File Section injection.

Linux UEFI Validation (LUV)

- https://01.org/linux-uefi-validation
- LUV (Linux UEFI Validation) is a Linux distribution that helps OEMs build UEFI systems. LUV includes CHIPSEC, FWTS, BITS, and a few other tools. It has unique tests that test firmware updates over multiple reboots. LUV is based on Yocto Linux, and works on Intel x86 and x64 systems; Linaro is porting LUV to ARM AArch64. LUV-live is a LUV-based liveboot distribution. It boots via a thumbdrive or via PXE, and runs in batch mode and gathers up test results.

Tianocore

- http://www.tianocore.org/
- Tianocore is the codename name of the UEFI Forum's BSD-licensed open source implementation of UEFI, the infrastructure code that many vendors use, along with all of their own unique code. It includes multiple developer tools to create and manipulate UEFI containers. It works on MacOSX, Windows, or Linux. It works on Intel, AMD, and ARM systems. It works with multiple C compiler toolchains, GCC, ICC, MSC, LLVM Clang. The EDK2 (EFI Development Kit V2) includes a UEFI emulator. It also includes a UEFI emulator. It includes an EDK2 (EFI Development Kit V2), for writing UEFI drivers. The EDK2 is a trunk, there are multiple branches, including one that uses clang's analysis and security abilities. For those that don't follow trunk, here are periodic snapshot releases of the EDK2 trunk, called UDK<year>, (Uefi Development Kit), where <year> is the year of the last revision date of the UEFI specifications.

UEFI Shell

- The UEFI Shell and it's commands are somewhat like the MS-DOS command interpreter and it's commands. And it's also somewhat like a kernel debugger, since the shell was created by EFI developers to test their code.
- Some of the UEFI Shell's console commands: Alias, Attrib, Bcfg, CD, CLS, Comp, Connect, CP, Date, Dblk, Devices, DevTree, DH, Disconnect, DMem, DmpStore, DP, Drivers, DrvCfg, DrvDiag, EfiCompress, EfiDecompress, GetMTC, GoTo, Help, IfConfig, Load, LoadPCIROM, LS, Map, MemMap, MkDir, MM, Mode, MV, OpenInfo, Parse, Pause, PCI, Ping, Reconnect, Reset, RM, SerMode, Set, SetSize, SetVar, Shift, SMBIOSView, Stall, Time, TimeZone, Touch, Type, Unload, Ver, Vol
  (and there are a few newer commands, too.)
- UEFI Shell's full-screen commands: Edit, HexEdit
- UEFI Shell's scripting commands: Echo, Else, EndFor, EndIf, Exit, For, If,

Vim ported to UEFI

- https://github.com/mischief/efivim
- Because the UEFI Shell's “edit.efi” command is about as powerful as MS-DOS's “edit.com”.
- (No, AFAIK, there is no UEFI port of Emacs.)

CPython for UEFI

- edk2/AppPkg/Applications/Python/
- Intel has ported CPython 2.7x (not 3.x) to UEFI. The patch is included in Tianocore's EADK. This means you can write python scripts that run inside the UEFI Shell! Unless you have existing experience with UEFI Shell scripting language, Python scripts might be easier than writing UEFI Shell batch files.
- CHIPSEC bundles CPython binaries. Use the EDK2 to build CPython from source, it includes UEFI patches to python.org's sources.

QEMU / OVMF/AVMF

- http://qemu.org/
- QEMU is a popular emulator. Intel has “Open Virtual Machine Format” (OVMF). ARM has “AVMF” variant. This defines the virtualized UEFI forum model. Tianocore has a virtual target of UEFI that uses QEMU and OVMF/AVMF, useful to test loading the firmware and OS handover. Linaro (ARM Ltd's) has a fork of QEMU that has latest ARM-centric UEFI issues.

Visual UEFI

- https://github.com/ionescu007/VisualUefi
- VisualUEFI is a Solution and set of Visual Studio 2015 Project Files to allow building the official EDK-II without the use of inf files, python and 50 other build tools, a custom dependency tracker and build system, and twenty other custom pieces of code. The EDK-II is present as a submodule, directly from the   official TianoCore Tree, and no changes are done to it. It has show two UEFI sample components: A UEFI Application, and a UEFI Boot Driver. The code is EDK-II compatible, but built with VisualUEFI instead. Visual UEFI also includes a working copy of QEMU64 2.7 for Windows, with a fairly recent UEFI 2.6 OVMF Secure Boot ROM.
- If you use the Tianocore command line tools, but prefer an Visual Studio GUI, this is your tool.

Eclipse EDK2 plugin

- https://github.com/ffmmjj/uefi_edk2_wizards_plugin
- This project is an Eclipse plugin that aims to provide a set of Eclipse wizards on top of Eclipse CDT to ease the development of EDK2-based UEFI modules.
- If you use the Tianocore command line tools, but prefer an Eclipse GUI, this is your tool.

CrScreenshotDxe

- https://github.com/LongSoft/CrScreenshotDxe
- CrScreenshotDxe is a UEFI DXE driver to take screenshots from GOP-compatible graphic consoles. This DXE driver tries to register keyboard shortcut (LCtrl + LAlt + F12) handler for all text input devices. The handler tries to find a writable FS, enumerates all GOP-capable video devices, takes screenshots from them and saves the result as PNG files on that writable FS. The goal is to be able to make BIOS Setup screenshots for systems without serial console redirection support, but it can also be used to take screenshot from UEFI shell, UEFI apps and UEFI bootloaders.

DarwinDumper

- https://bitbucket.org/blackosx/darwindumper
- DarwinDumper is collection of open source tools to dump Apple Mac OS X system information to aid troubleshooting. It dumps ACPI tables, DMI, EFI memory, EFI variables, SMC keys, system BIOS, etc. It has a privacy mode which omits some serial numbers and machine-unique data from the resulting report.
- Tools include: bdmesg, cmosDumperForOsx, dmidecode, dumpACPI, edid-decode, fdisk440, FirmwareMemoryMap, flashrom, getcodecid, genconfig, getdump, gfxutil, iasl, ioregwv, lzma, nvram, oclinfo, lspci, RadeonDump, radeon_bios_decode, smbios-reader, SMC_util, VoodooHDA.kext, x86info.

UEFI Utiliities

- https://github.com/fpmurphy/UEFI-Utilities-2016
- FPMurphy's UEFI Utilities is a collection of command line tools that dumps information. Tools include: DisplayBMP, ScreenModes ShowBGRT, ShowECT, ShowEDID, ShowESRT, ShowMSDM, ShowOSIndication, ShowRNG, ShowTCM20, ShowTPM2, ShowTrEE, and ShowTrEELog.

FlashROM

- https://www.flashrom.org/Flashrom
- https://github.com/pinczakko/winflashrom
- flashrom is an open source utility for identifying, reading, writing, verifying and erasing flash chips. It is designed to flash BIOS/EFI/coreboot/firmware/optionROM images on mainboards, network/graphics/storage controller cards, and various other programmer devices. It supports parallel, LPC, FWH and SPI flash interfaces and various chip packages. It works on most Unix system, and there is a Windows port.

ACPIdump

- https://www.acpica.org/source
- The ACPI Component Architecture (ACPICA) project provides an operating system (OS)-independent reference implementation of ACPI. The complexity of the ACPI specification leads to a lengthy and difficult implementation in operating system software. The primary purpose of ACPICA is to simplify ACPI implementations for OSVs by providing major portions of an ACPI implementation in OS-independent ACPI modules that can be integrated into any OS.
- ACPICA includes a tool called ACPIdump. This tool works on multiple OSes, as well as having a native UEFI port.

UEFIreverse

- https://github.com/jethrogb/uefireverse
- UEFIreverse is a collection of UEFI reverse engineering tools. This is a collection of tools to help reverse UEFI-based firmware. It includes:
  - efiperun - Load and run EFI PE image files on your favorite operation system (Linux). See efiperun/README.md for more information.
  - guiddb - Scan UEFI source build files (.DEC files) for GUIDs and output them in C-source file format. Includes A database of known guids.
  - memdmp - A patched version of the UEFI Shell's MemMap command that creates a memory dump file. Then, run dmp2seg to convert that output file into many files with the actual memory contents. Then, run make_elf.rb to make a single ELF file with all the memory contents. The ELF file is not executable or anything, it's just a convenient format to store memory segments.
  - tree: A class file that will provides a Ruby tree abstraction for a firmware tree on your filesystem previously extracted by UEFITool's UEFIExtract.

Read and Write Everything

- http://rweverything.com/
- RW, aka RWEverything (Read and Write Everything) is a GUI Windows-based firmware utility that enables access to almost all the computer hardware, including PCI (PCI Express), PCI Index/Data, Memory, Memory Index/Data, I/O Space, I/O Index/Data, Super I/O, Clock Generator, DIMM SPD, SMBus Device, CPU MSR Registers, ATA/ATAPI Identify Data, Disk Read Write, ACPI Tables Dump (include AML decode), Embedded Controller, USB Information, SMBIOS Structures, PCI Option ROMs, MP Configuration Table, E820, EDID and Remote Access. It ships with Win32 or Win64 binaries, and is freeware, not open source.

Read Universal utility

- http://ruexe.blogspot.tw/
- The Read Universal utility is a multi-function tool for BIOS debugging. It includes tools that provides direct access to almost all resources like memory, IO space, PCI, SMBIOS data, UEFI variables and so on. The tool is freeware -- not open source - and is written by a UEFI Engineer at AMI. It ships as ru.exe and ru.efi binaries and is freeware, not open source.

UBU and UBU-Helpers

- http://www.win-raid.com/t154f16-Tool-Guide-News-quot-UEFI-BIOS-Updater-quot-UBU.html
- https://github.com/LongSoft/UBU-helpers
- The UEFI BIOS Updater (UBU) detects the versions of the OptionROM/EFI modules, which are inside an AMI UEFI BIOS file and to update:the most important OROM/EFI modules and the CPU MicroCode of any AMI Aptio IV UEFI BIOS. It is used in the firmware modding community. UBU is freeware, not open source.
- In addition to the UBU freeware tool, there's a related project called UBU-Helpers, which is open source. UBU-helpers is a collection of tools used to examine UEFI binaries. There are three tools: Hex Find, Find Version, and Driver Version. HexFind searches for a pattern in a file. FindVer searches for version-based patterns in a file. DriVer searches for multiple UEFI drivers/applications via hardcoded string/offset searches. UBU-helpers can be used without using UBU.

Malware POCs from Cr4sh

- ThinkPwn, https://github.com/Cr4sh/ThinkPwn
- FwExpl, https://github.com/Cr4sh/fwexpl
- PeiBackdoor, https://github.com/Cr4sh/PeiBackdoor
- SmmBackdoor, https://github.com/Cr4sh/SmmBackdoor

Commercial Intel/ARM tools/devices

- Intel:
  - Intel Tunnel Mountain box, http://tunnelmountain.net/
  - Intel Minnowboard, http://minnowboard.org/
  - Intel System Studio, https://software.intel.com/en-us/intel-system-studio
- ARM:
  - ARM Juno AArch64 dev board
  - ARM Developer Studio DS-5, https://developer.arm.com/products/software-development-tools/ds-5-development-studio

Questions?

- Questions?
- Comments?
- What are the main tools that I missed?
- Slides will be posted online soon. Look for a blog post on FirmwareSecurity.com.
- I'll be at the conference all day, speak up if you want to see some Linux demos of these tools.
- Thanks!