

Cyber Strike Solutions, LLC

<!--Securing computers one byte at a time-->

CHIP-LEVEL VULNERABILITY ASSESSMENT USING CHIPSEC AND LUVOS

Dr. Jeffrey L. Struik, CISSP, PenTest+



The Agenda



- Introduction
- What is CHIPSEC
- What is LuvOS
- Demonstration
- Conclusion



Introduction

- Married over 17 years, 6 kids, and 2 dogs named Linux and Zenith
- 10 years in the U.S. Army as Apache electrician
- ~10 years cyber security experience
- Primary experience with DoD customers
- NIST Risk Management Framework
- Owns Cyber Strike Solutions, LLC



Why discuss chip-level/firmware security

- Low level breach nearly impossible to detect
- Attackers gain irrevocable persistence on the device
- Could prevent usability of device
- Devices are everywhere
 - Thanks IoT
- Learning new things is fun



This Photo by Unknown Author is licensed under CC BY-NC-ND



Cyber Strike Solutions, LLC

<!--Securing computers one byte at a time-->

CHIPSEC – What is it?



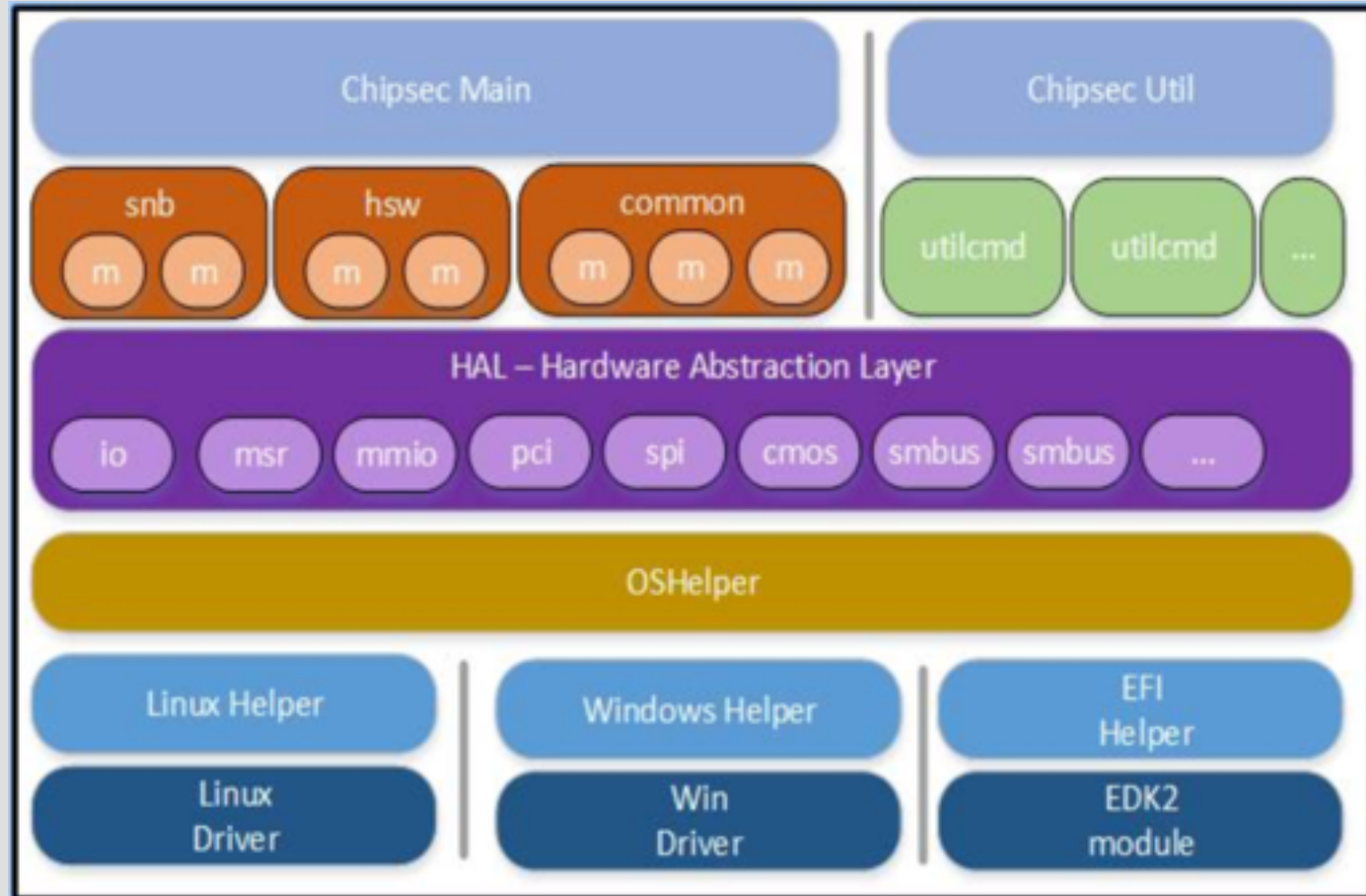
CHIPSEC

- Python and C based tool
- Checks chip level settings to validate proper configuration
- Can be used as a python module
- Runs on Linux, Windows, and Mac
- Executable in pre-boot environment via UEFI shell
 - Prevents the need to install additional software on OS
 - Prevents the need to alter configuration of the system



CHIPSEC - Structure

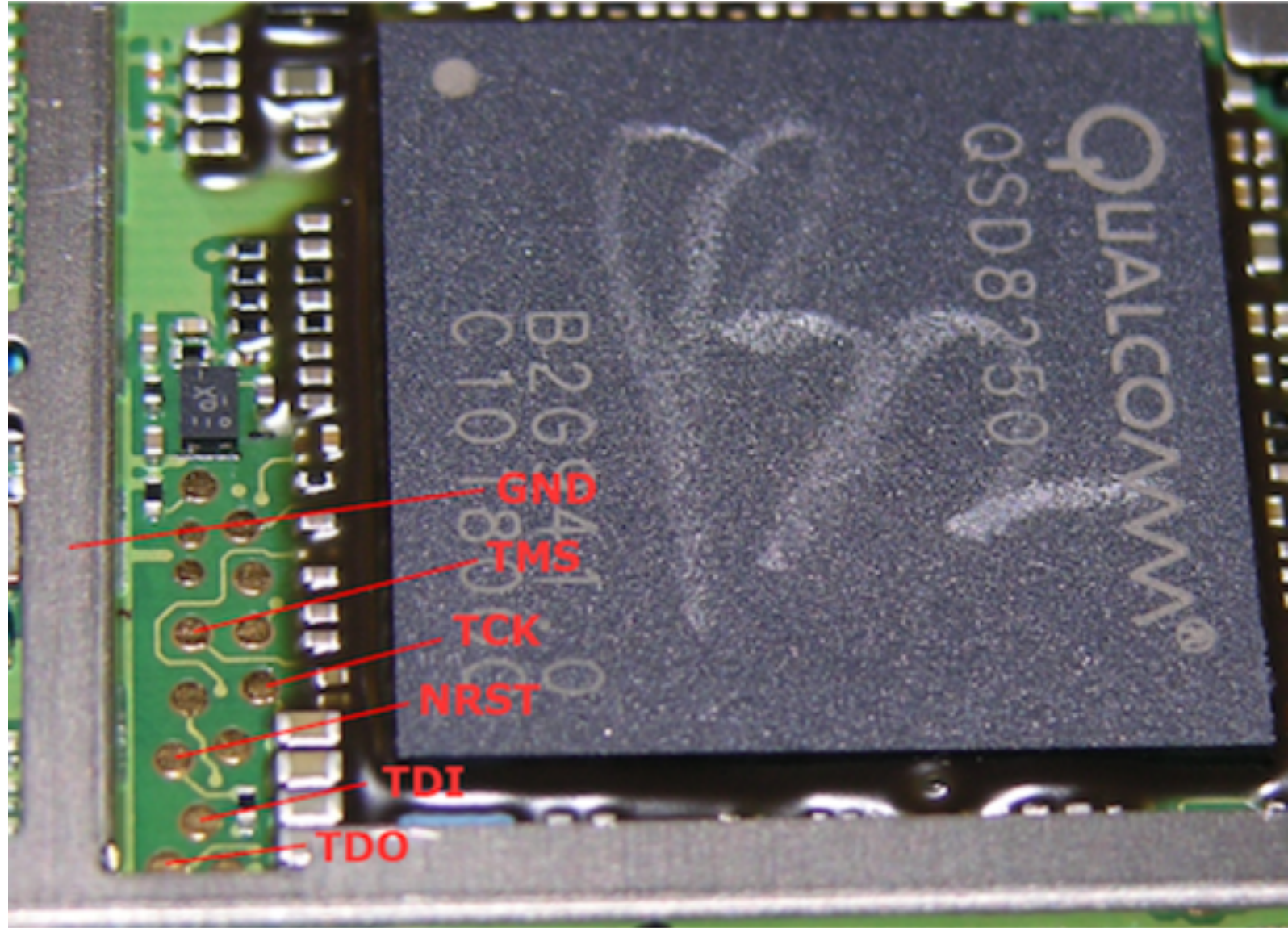
- Main and utility module
- Specific drivers and helpers per OS
- Common sets of components
- Hardware Abstraction Layer modules
- Specific test modules



CHIPSEC

Main Module

- Used to execute full test suite
- Allows specific module calls
- Multiple output formats



This Photo by Unknown Author is licensed under [CC BY-NC-ND](#)

CHIPSEC Main



- Provides valuable output about chip-level vulnerabilities
- Multiple output formats
 - Log File with verbose output
 - JSON with simple pass/fail
 - XML with verbose output
- Detect misconfigurations of board-level memories

CHIPSEC – Example tests-

```
[*] Keyboard buffer contents (at 0x41E):
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
[*] Checking contents of the keyboard buffer..

[+] PASSED: Keyboard buffer looks empty. Pre-boot passwords don't seem to be exposed

[CHIPSEC] ***** SUMMARY *****
[CHIPSEC] Time elapsed          0.001
[CHIPSEC] Modules total        1
[CHIPSEC] Modules failed to run 0:
[CHIPSEC] Modules passed       1:
[+] PASSED: chipsec.modules.common.bios_kbrd_buffer
[CHIPSEC] Modules information  0:
```

- Command executed – `chipsec_main -m common.bios_kbrd_buffer`
- Performs test to verify BIOS passwords are non-existent in the keyboard buffer

CHIPSEC – S3 Boot Script

```
[ - ] Dispatch opcode (off 0x42FF) with entry-point 0x000000009CE75260 > UNPROTECTED
[*] Found 58 Dispatch opcodes
[ - ] Entry-points of Dispatch opcodes in S3 boot-script are not in protected memory

[ - ] FAILED: S3 Boot-Script and Dispatch entry-points do not appear to be protected
[!] Additional testing of the S3 boot-script can be done using tools.uefi.s3script_modify

[CHIPSEC] ***** SUMMARY *****
[CHIPSEC] Time elapsed          0.234
[CHIPSEC] Modules total        1
[CHIPSEC] Modules failed to run 0:
[CHIPSEC] Modules passed       0:
[CHIPSEC] Modules information   0:
[CHIPSEC] Modules failed       1:
[ - ] FAILED: chipsec.modules.common.uefi.s3bootscript
```


CHIPSEC – S3 Script Modify

```
S3 Boot Script AA Parser
[*] S3 boot script type: 0xAA
[*] Looking for TERMINATE opcode in the script at 0x000000009CDF2000..
[+] Found TERMINATE opcode at offset 0x4741
Entry at offset 0x4741 (len = 0x3, header len = 0x0):
Data:
ff 00 03 | 00 03
Decoded:
Opcode      : S3_BOOTSCRIPT_TERMINATE (0xFF)

[mem] 0x000000009CDF6741
[*] New S3 boot script opcode:
Opcode      : S3_BOOTSCRIPT_DISPATCH (0x0008)
Entry Point: 0x00000000250934000

[*] Adding new opcode entry at address 0x000000009CDF6741..
08 00 0b 00 40 93 50 02 00 00 00 | @ P
[mem] buffer len = 0xB to PA = 0x000000009CDF6741
08 00 0b 00 40 93 50 02 00 00 00 | @ P
[*] Moving TERMINATE opcode to the last entry at 0x000000009CDF674C..
[mem] buffer len = 0x3 to PA = 0x000000009CDF674C
ff 00 03 |
[+] PASSED: The script has been modified. Go to sleep..
```

- Demonstrates vulnerable configuration of S3 Bootscript
- Script successfully added new opcode
- Moved TERMINATE opcode to a different memory address



CHIPSEC Utility

WARNING : chipsec_utility allows DIRECT access to hardware!!!!

- Plethora of scripts to gather additional hardware information
- Dump memory
 - Binaries
 - ROMS
 - Variables
 - Keys



[This Photo](#) by Unknown Author is licensed under [CC BY-SA-NC](#)

CHIPSEC Utility Scripts



This Photo by Unknown Author is licensed under [CC BY-SA](#)

- PCI command – enumerate and direct access to PCI config registers
- TPM command – enumerate various TPM variables
- UCODE – provides current microcode ID for the CPU, and load updates



CHIPSEC – Live Demo

CHIPSEC - Recap



[This Photo](#) by Unknown Author is licensed under [CC BY](#)

- Provides wide range of pre-configured vulnerability checks
- Allows for custom module creation
- Works on Mac, Linux, Windows, and custom UEFI shell
- Powerful tool – use with caution



Cyber Strike Solutions, LLC

<!--Securing computers one byte at a time-->

Linux UEFI
Validation OS



Linux UEFI Validation OS (LuvOS)

- Open Source project backed by Intel
- Live Operating System
- Swiss Army knife of low level system testing
- Produces packaged reports



[This Photo](#) by Unknown Author is licensed under [CC BY](#)



Linux UEFI Validation Tools

- BITS – BIOS Implementation Test Suite
- NDCTL – Tool for managing LIBNVDIMM kernel subsystem
- FWTS – FirmWare Test Suite
- Kernel-EFI-Warnings – Tool to look at warnings produced by the kernel
- Pstore_tests – Specific to Linux – designed to record data at kernel crash
- CHIPSEC



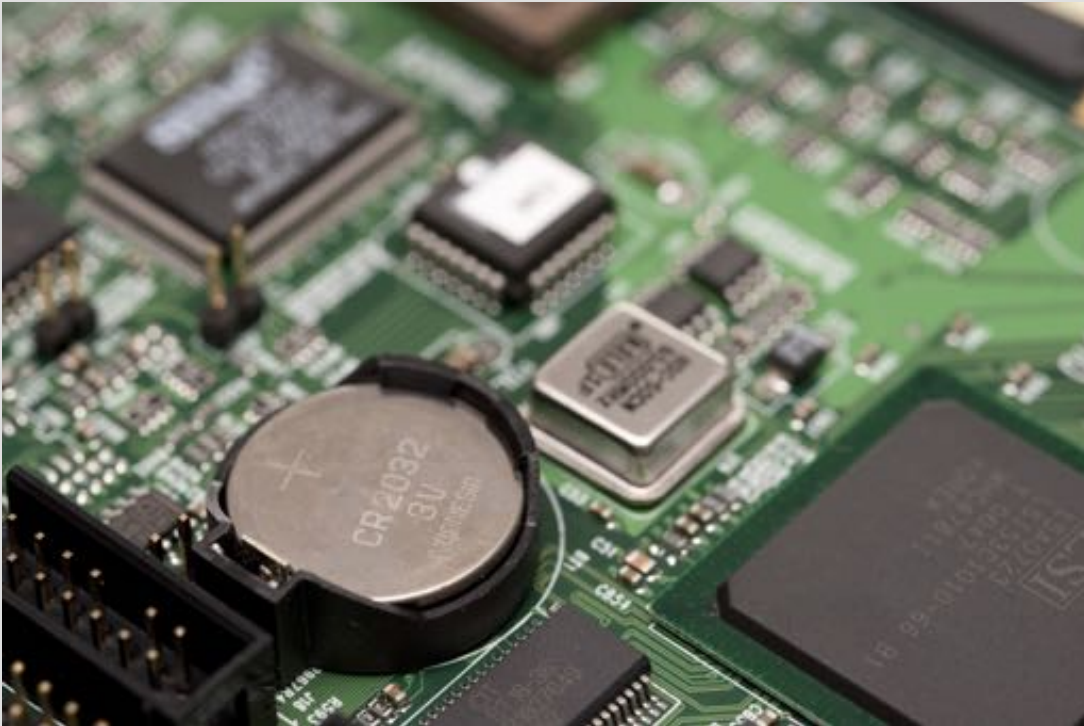
Linux UEFI Validation Platform

- Use Yocto as the base platform
- Three different configurations
 - Netboot 32/64 bit
 - USB boot MBR 32/64 bit
 - USB boot GPT 32/64 bit



[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)

Linux UEFI Validation (LuvOS) - BITS



[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)

- BIOS Implementation Test Suite
 - Allows checking of microcode
 - Supports Python scripting
 - LuvOS scripts the BITS test
 - Can additionally be run from grub
 - Uses Python for execution

BITS Executing

```
Summary: 8 passed, 0 failed
---- ACPI _PSS (Pstate) runtime tests ----
[assert] P0: Turbo measured frequency 1900 >= expected 1901 MHz FAIL
[assert] P13: measured frequency 800 MHz == expected 779 MHz FAIL
Summary: 13 passed, 2 failed
---- ACPI DSDT (Differentiated System Description Table) ----
Summary: 1 passed, 0 failed
---- ACPI FACP (Fixed ACPI Description Table) ----
Summary: 1 passed, 0 failed
---- ACPI HPET (High Precision Event Timer Table) ----
Summary: 1 passed, 0 failed
---- ACPI MADT (Multiple APIC Description Table) ----
Summary: 1 passed, 0 failed
---- ACPI MPST (Memory Power State Table) ----
Summary: 0 passed, 0 failed
---- ACPI RSDP (Root System Description Pointer Structure) ----
Summary: 2 passed, 0 failed
---- ACPI XSDT (Extended System Description Table) ----
Summary: 1 passed, 0 failed

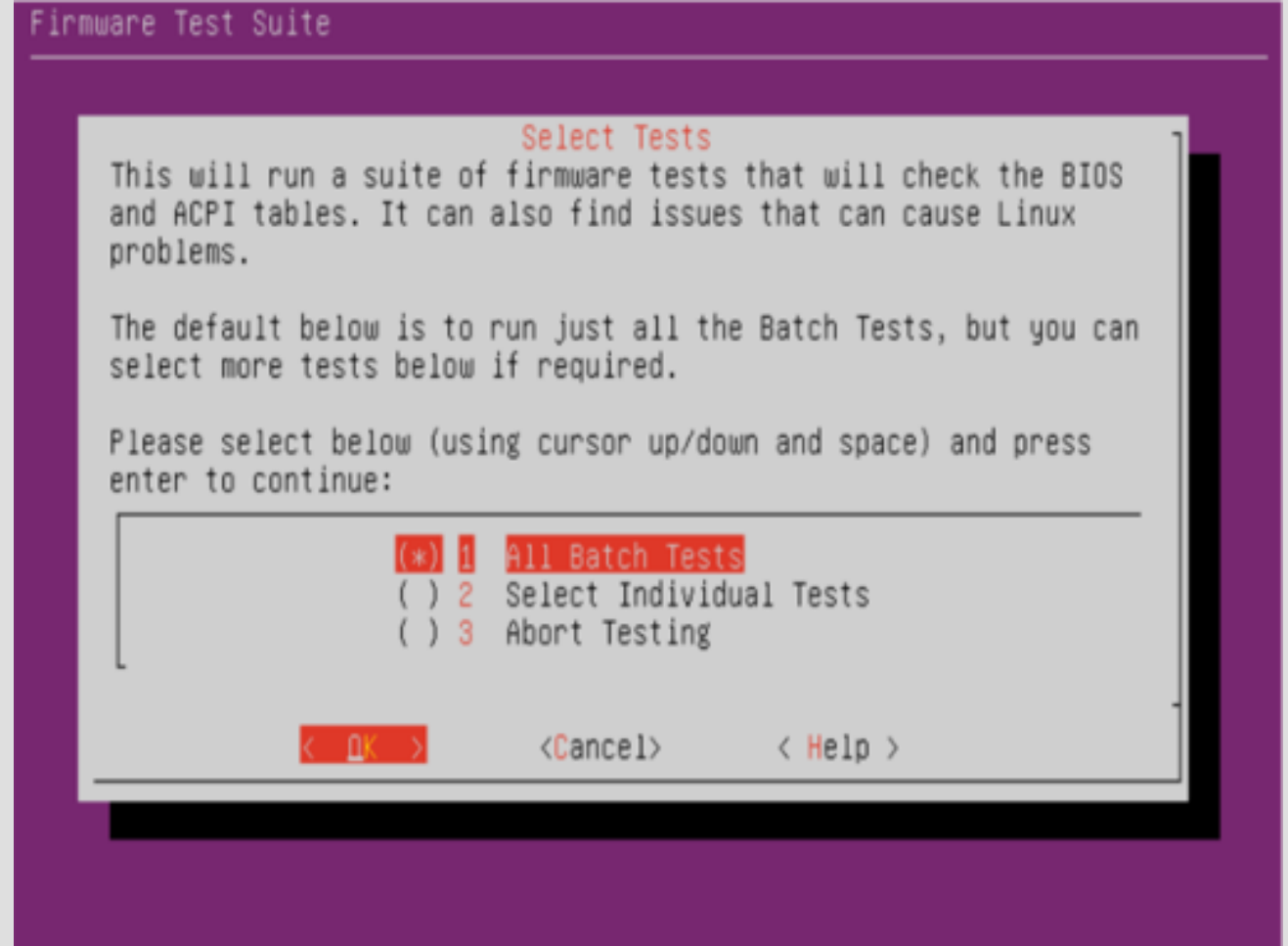
==== EFI Tests ====
---- EFI System Table CRC ----
Summary: 1 passed, 0 failed
---- EFI Runtime Services Table CRC ----
Summary: 1 passed, 0 failed
```

- Executes before LuvOS boots up



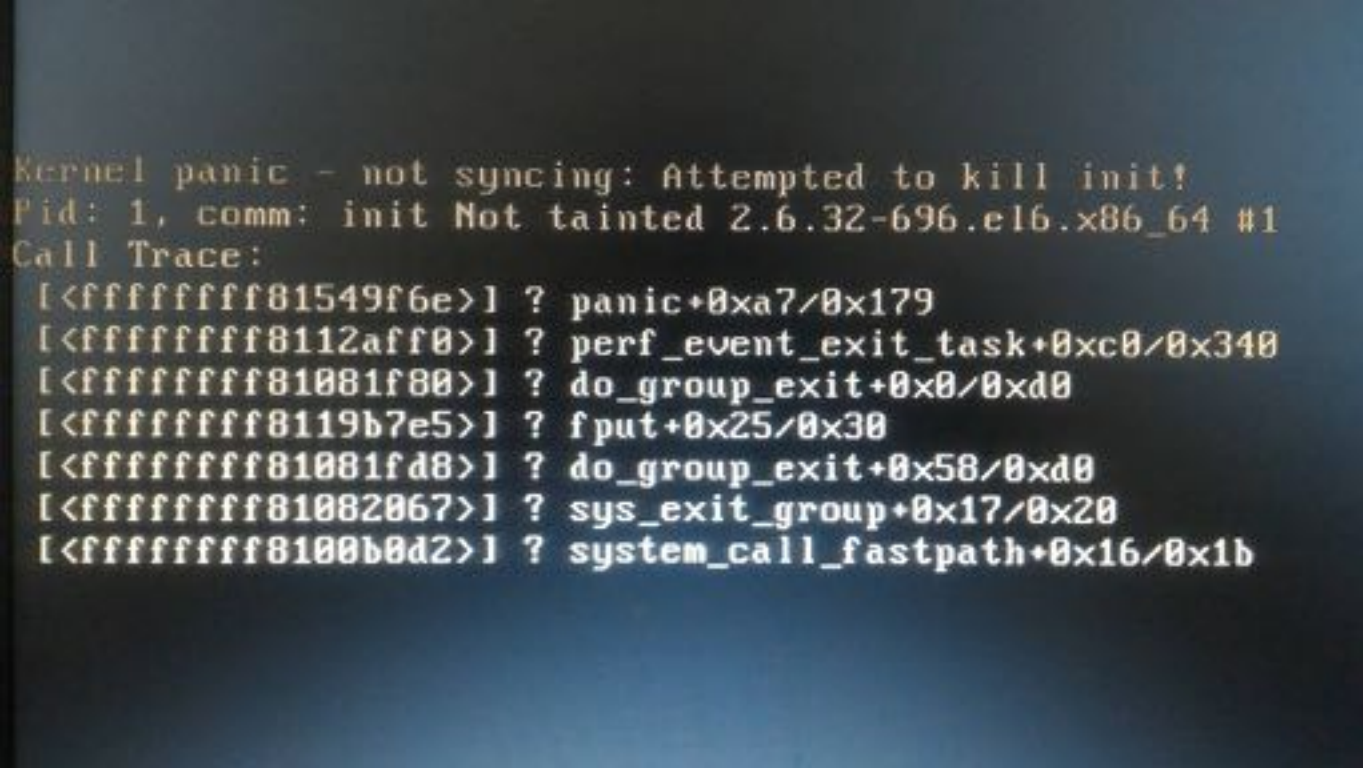
LuvOS – Firmware Test Suite (FWTS)

- Can act as a stand-alone pre-boot environment
- Supposed to identify BIOS/UEFI/ACPI errors
- Can run in batch mode or specific tests mode
- Over 50 tests
- Executes within Linux, Live-OS, and/or LuvOS
- Some overlaps with CHIPSEC



This Photo by Unknown Author is licensed under [CC BY-SA-NC](#)

Kernel-EFI-Warnings



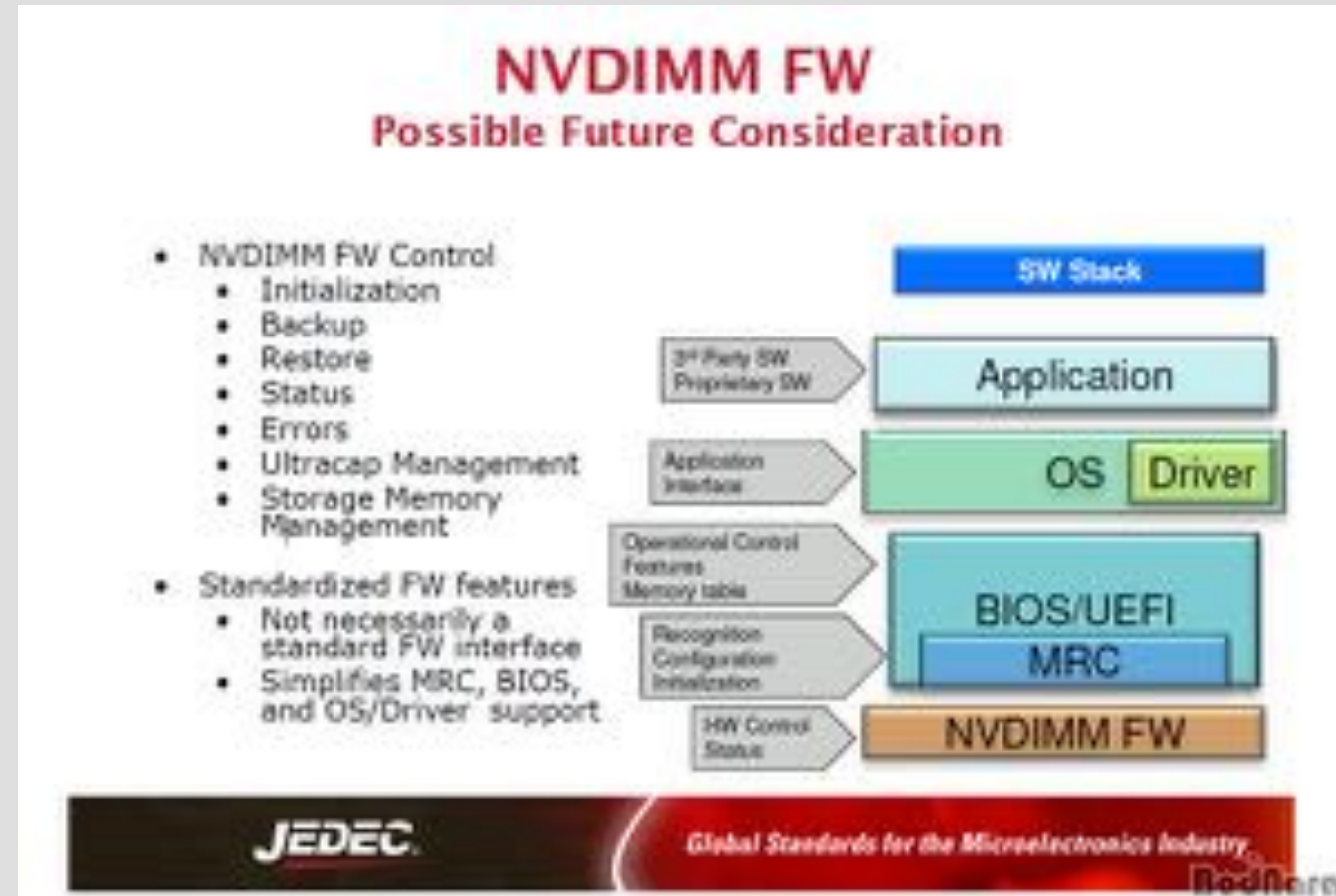
```
Kernel panic - not syncing: Attempted to kill init!  
Pid: 1, comm: init Not tainted 2.6.32-696.el6.x86_64 #1  
Call Trace:  
[<ffffffff81549f6e>] ? panic+0xa7/0x179  
[<ffffffff8112aff0>] ? perf_event_exit_task+0xc0/0x340  
[<ffffffff81081f80>] ? do_group_exit+0x0/0xd0  
[<ffffffff8119b7e5>] ? fput+0x25/0x30  
[<ffffffff81081fd8>] ? do_group_exit+0x58/0xd0  
[<ffffffff81082067>] ? sys_exit_group+0x17/0x20  
[<ffffffff8100b0d2>] ? system_call_fastpath+0x16/0x1b
```

- Shell script
- Analyzes dmesg buffer
- Looks for kernel messages indicative of possible UEFI bugs

[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)

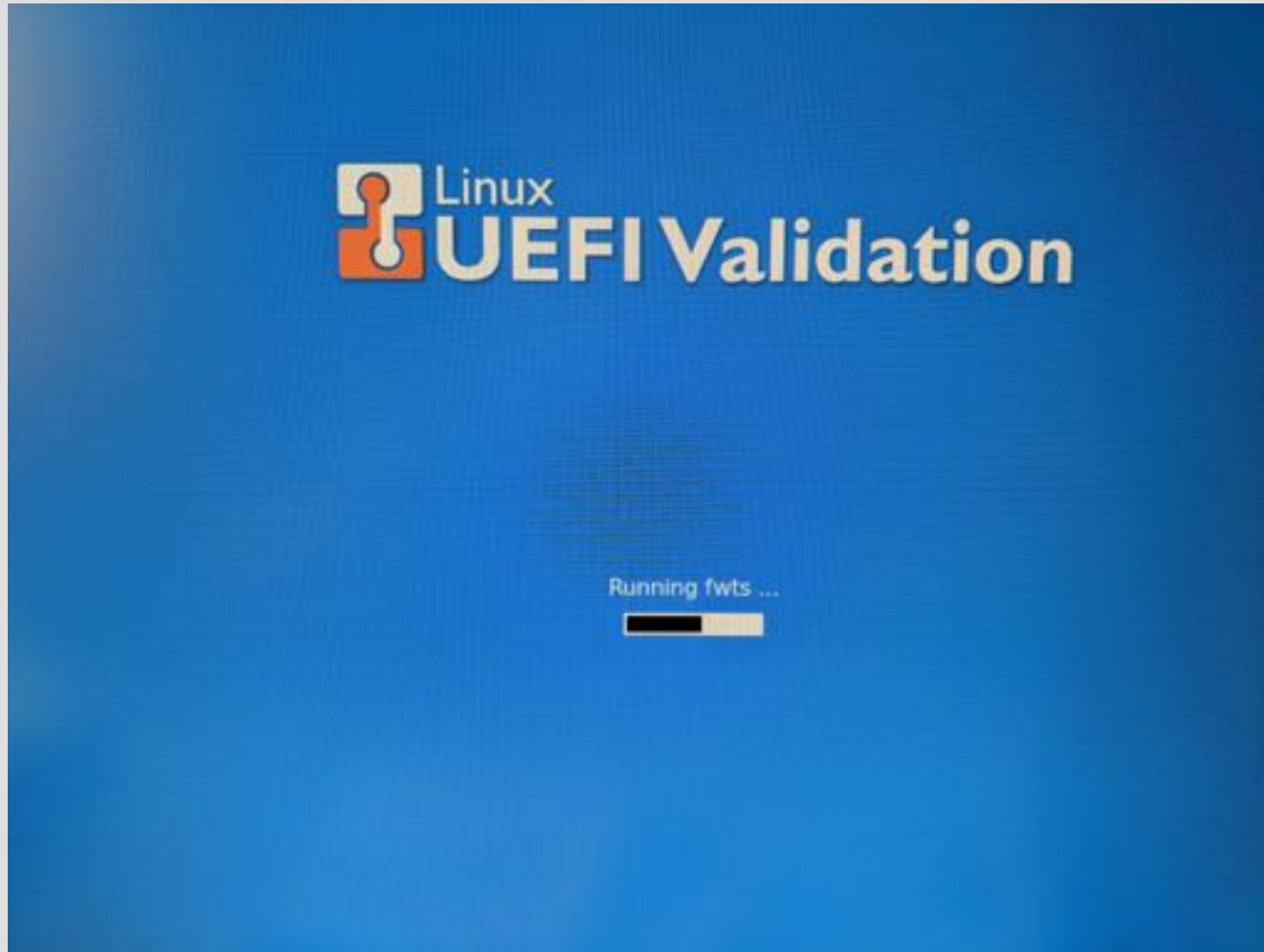
NDCTL

- Tests system NVDIMM
- Checks security configuration
- Checks additional NVDIMM settings
 - Namespace create/destroy
 - Memory configuration

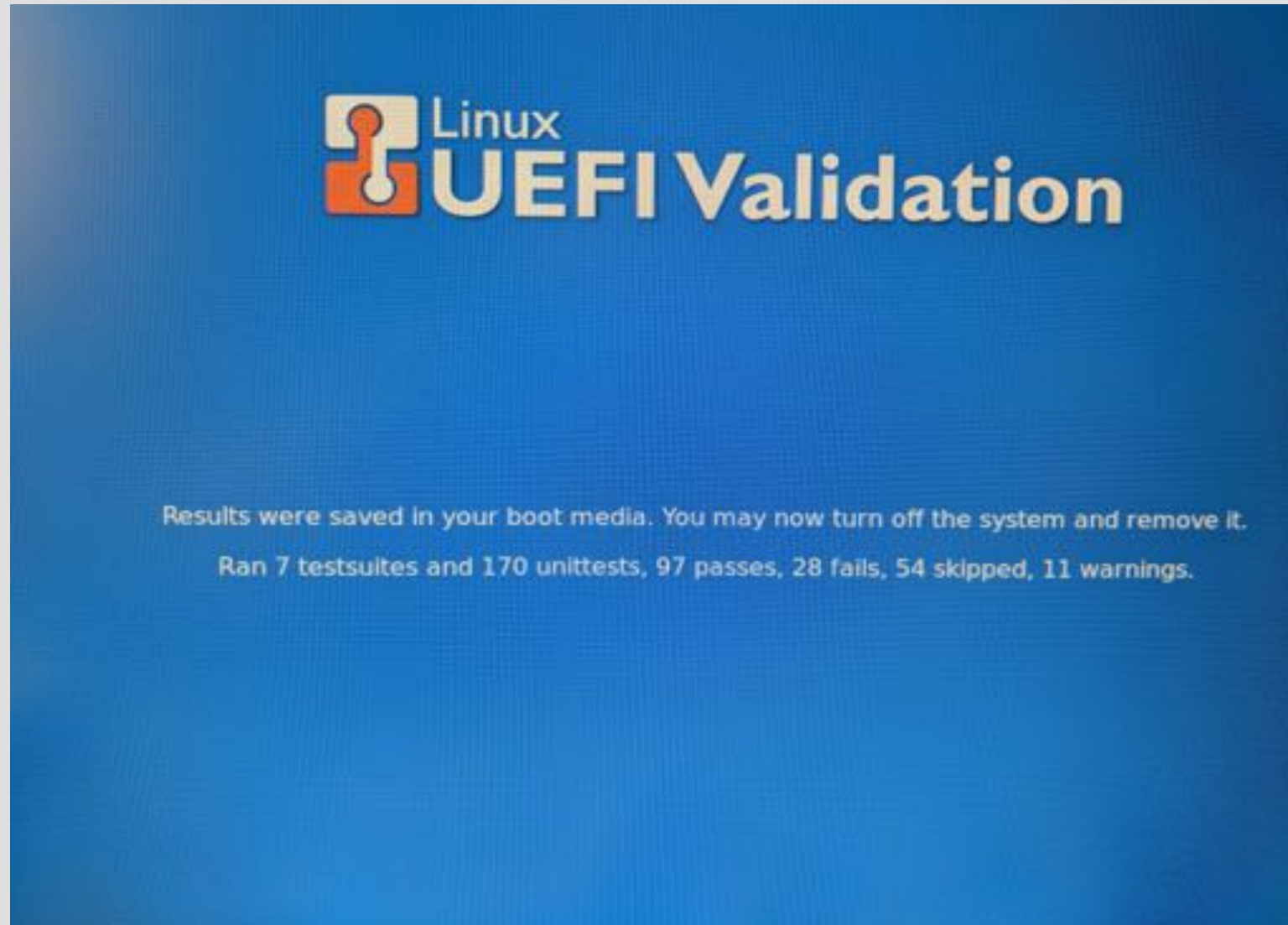


This Photo by Unknown Author is licensed under [CC BY-NC-ND](#)

LuvOS During Execution



LuvOS During Execution



LuvOS Results - Example

- luv-results-2019-03-30--17-34-59/luv.html



Cyber Strike Solutions, LLC

<!--Securing computers one byte at a time-->

Conclusion



Conclusion

- Proliferation of computing devices means chips/firmware are everywhere
- CHIPSEC is a powerful tool
 - Ability to perform vulnerability assessment of board level memory
 - Use Utility script to dig deeper
 - Be careful – modifications could brick your device



Conclusion

Linux UEFI Validation OS

- Provides several tools to test a system
- Includes CHIPSEC
- Can be used to PXE boot for increased efficiency
- Provides HTML report of findings



[This Photo](#) by Unknown Author is licensed under [CC BY](#)

Questions



[This Photo](#) by Unknown Author is licensed under [CC BY-NC](#)



[This Photo](#) by Unknown Author is licensed under [CC BY](#)