CHIPSEC Quick Reference for System Administrators

by PreOS Security

v0.4, 2017-07-30

CHIPSEC, by Intel Advanced Threat Research, is a firmware diagnostic and vulnerability assessment tool for 32-/64-bit Intel BIOS/UEFI systems running Linux, Windows, macOS, or UEFI Shell. This reference shows a subset of it's defensive commands, for initial system testing. Read the CHIPSEC manual & wiki pages *https://github.com/chipsec/* for install info and additional commands.

Once CHIPSEC is installed, CPython 2.7x has two new modules to call, *chipsec_main* and *chipsec_util. The initial use of CHIPSEC*:

chipsec_main

This will run all available security tests relevant for this system. To run other specific commands:

chipsec_main -m <*tool*> <*...*>
chipsec_util -m <*utility*> <*...*>

CHIPSEC_Main Command Line Options:

-m --module <m>: Specify module <m> to run.
-a --module_args <a>: Additional module arguments.
-n –no_driver: Don't load the OS kernel driver. (limits the tool set to non-driver modules)
-i –ignore_platform: Try to run on unsupported platform.
-v –verbose: Verbose mode.
-d –debug: Show debug output.
-l --log <f>: Output using ASCII.
-j --json <f>: Output using JSON.
-x --xml <f>: Output using XML.
-t --moduletype <t>: Run tests of a specific tag type.
--list_tags: List all the available module tags.
-I --include <p>: Specify additional path to load modules.
--failfast: Fail on any exception and exit.
--no_time: Don't log timestamps.
-p –platform: Specify 3-character platform code: AVN (Avoton), BDW (Broadwell), BYT (Bay Trail), CHT (Cherry Trail), Braswell), HSW (Haswell), HSX (Haswell Server), IVT (Ivytown, Ivy Bridge-E), JKT (Jaketown, Sandy Bridge-E), KBL (Kaby Lake), QRK (Quark), SKL (Skylake).

CHIPSEC Main Tests:

memconfig: Verify memory map registers are correctly configured.
remap: Verify memory remapping configuration.
smm_dma: Examines SMRAM configuration for DMA attacks.
common.secureboot.variables: Verify the UEFI Secure Boot-related variables are protected.
common.uefi.access_uefispec: Verify the protection of UEFI variables.
common.uefi.s3bootscript: Check S3 Resume Boot-Script protections.
common.bios_kbrd_buffer: Checks BIOS/HDD password exposure via keyboard buffer.
common.bios_smi: Checks SMI event configurations.
common.bios_ts: Checks BIOS Interface Lock, including Top Swap Mode.
common.bios_wp: Checks BIOS Region Write Protection.
common.ia32cfg: Tests that IA-32/IA-64 features are configured and locked.
common.rtclock: Checks for RTC memory locks.
common.smm: Checks SMM memory (SMRAM) protection.
common.smrr: Checks for CPU SMM Cache Poisoning and SMMs are enabled and configured.
common.spi_desc: Checks that unauthorized software is unable to write to the SPI Flash Descriptor.
common.spi_fdopss: Checks for SPI Controller Flash Descriptor Security Override Pin Strap.
common.spi_lock: Checks if SPI Flash Controller Configuration is locked.

CHIPSEC Main Tools:

tools.secureboot.te, tools.cpu.sinkhole, tools.smm.smm_ptr, tools.uefi.blacklist, tools.uefi.s3script_modify, tools.vmm.cpuid_fuzz, tools.vmm.hypercallfuzz, tools.vmm.iofuzz, tools.vmm.msr_fuzz, tools.vmm.pcie_fuzz, tools.vmm.pcie_overlap_fuzz, tools.vmm.venom, tools.vmm.hv.hypercallfuzz, tools.vmm.hv.synth_dev, tools.vmm.hv.synth_kbd, tools.vmm.hv.vmbusfuzz, tools.vmm.vbox.vbox_crash_apicbase, tools.vmm.xen.hypercallfuzz, tools.vmm.xen.xsa188

blacklist: Check for blacklisted UEFI executables.
chipsec_main -m tools.uefi.blacklist
chipsec_main [-i] [--no_driver] -m tools.uefi.blacklist [-a <*fw_image*>,<*blacklist*>]

whitelist: Check for whitelisted UEFI executables.
chipsec_main -m tools.uefi.whitelist [-a generate| check,<*json*>,<*fw_image*>]

CHIPSEC_Util Utilities:

acpi, cmos, cpu, decode, idt, gdt, ec, igd, io, iommu, ldt, mem, mmcfg, mmio, msgbus, msr, nmi, pci, platform, reg, smbus, smi, spd, spi, spidesc, ucode, uefi, vmm.

acpi: Provides access to ACPI tables.
chipsec_util -m acpi list
chipsec_util -m acpi table <*name*>|<*file_path*>
chipsec_util -m acpi table XSDT
chipsec_util -m acpi table acpi_table.bin

cpu: Display CPU information.
chipsec_util cpu info
chipsec_util cpu cr <*cpu_id*> <*cr_number*> *[value]*
chipsec_util cpu cpuid <*eax*> *[ecx]*
chipsec_util cpu pt *[paging_base_cr3]*

cmos: CMOS command.
chipsec_util cmos dump
chipsec_util cmos readl|writel|readh|writeh <*offset*> *[val]*
chipsec_util cmos rl 0x0
chipsec_util cmos wh 0x0 0xCC

decode: Decode a 'rom.bin' image file of a SPI flash dump (see SPI command).
chipsec_util -m decode <*rom*> *[fw_type]*
chipsec_util -m decode types: chipsec_util -m decode spi.bin vss

ec: Embedded Controller command.
chipsec_util ec index *[<offset>]*
chipsec_util ec dump *[<size>]*
chipsec_util ec command <*command*>
chipsec_util ec read <*start_offset*> *[<size>]*
chipsec_util ec write <*offset*> <*val*>

io: Allows direct access to read and write I/O port space.
chipsec_util io list
chipsec_util io <*io_port*> <*width*> *[value]*
chipsec_util io 0x61 1
chipsec_util io 0x430 byte 0x0

iommu: Provides access to I/O Memory Management Unit (IOMMU) engines, e.g. Intel VT-d. The 'pt' command dumps the IOMMU Page Tables.
chipsec_util iommu list
chipsec_util iommu config <*iommu_engine*>
chipsec_util iommu status <*iommu_engine*>
chipsec_util iommu enable|disable <*iommu_engine*>
chipsec_util iommu pt

Examples:
chipsec_util iommu config VTD
chipsec_util iommu status GFXVTD
chipsec_util iommu enable VTD

mem: Provides direct access to physical memory.
chipsec_util mem read|readval|write|writeval|allocate| pagedump <*physical_address*> <*length*> *[value|buffer_file]*
chipsec_util mem readval 0xFED40000 dword
chipsec_util mem read 0x41E 0x20 buffer.bin
chipsec_util mem writeval 0xA0000 dword 0x9090CCCC
chipsec_util mem write 0x100000000 0x1000 buffer.bin
chipsec_util mem write 0x100000000 0x10 000102030405060708090A0B0C0D0E0F

```
chipsec_util mem allocate 0x1000
chipsec_util mem pagedump 0xFED00000  0x100000
```

mmcfg: Provides access to the Memory Mapped PCIe Configuration Space.

chipsec_util mmcfg <*bus*> <*device*> <*function*> <*offset*> <*width*> *[value]*

```
chipsec_util mmcfg 0 0 0 0x88 4
chipsec_util mmcfg 0 0 0 0x88 byte 0x1A
chipsec_util mmcfg 0 0x1F 0 0xDC 1 0x1
chipsec_util mmcfg 0 0 0 0x98 dword 0x004E0040
```

mmio: Provides access to Memory Mapped I/O (MMIO).

chipsec_util mmio list
chipsec_util mmio dump <*name*>
chipsec_util mmio read <*name*> <*offset*> <*width*>
chipsec_util mmio write <*name*> <*offset*> <*width*> <*value*>

```
chipsec_util mmio dump MCHBAR
chipsec_util mmio read SPIBAR 0x74 0x4
chipsec_util mmio write SPIBAR 0x74 0x4 0xFFFF0000
```

pci: Enumerate PCI/PCIe devices and expansion ROMs and allow direct access to PCI configuration registers via bus/device/function.

chipsec_util -m pci enumerate
chipsec_util -m pci <*bus*> <*device*> <*function*> <*offset*> *[width] [value]*
chipsec_util -m pci dump *[<bus> <device> <function>]*
chipsec_util -m pci xrom *[<bus> <device> <function>] [xrom_address]*

platform: Detect Chipsec/CPU.

chipsec_util platform

spi: Access the SPI Flash Controller. The Dump command creates a 'rom.bin' by the Decode command. The SPI Write and SPI Erase commands are dangerous.

chipsec_util -m spi info
chipsec_util -m spi info|dump|read|write|erase|disable-wp *[flash_address] [length] [file]*

```
chipsec_util -m spi dump rom.bin
```

spidesc: Parses a file containing a SPI Flash Descriptor.

chipsec_util spidesc *[rom]*

```
chipsec_util spidesc spi.bin
```

spd: SPD command.

chipsec_util spd detect
chipsec_util spd dump *[device_addr]*
chipsec_util spd read <*device_addr*> <*offset*>
chipsec_util spd write <*device_addr*> <*offset*> <*byte_val*>

```
chipsec_util spd dump DIMM0
chipsec_util spd read  0xA0 0x0
chipsec_util spd write 0xA0 0x0 0xAA
```

uefi: Provides access to UEFI variables, keys, and NVRAM.

chipsec_util -m uefi types
chipsec_util -m uefi var-list
chipsec_util -m uefi var-find <*name*>|<*GUID*>
chipsec_util -m uefi var-read|var-write|var-delete <*name*> <*GUID*> <*efi_variable_file*>
chipsec_util -m uefi decode <*rom_file*> *[fwtype]*
chipsec_util -m uefi nvram[-auth] <*rom_file*> *[fwtype]*
chipsec_util -m uefi keys <*keyvar_file*>
chipsec_util -m uefi tables
chipsec_util -m uefi s3bootscript *[script_address]*
chipsec_util -m uefi assemble <*guid*> freeform none|lzma|tiano <*raw_file*> <*uefi_file*>
chipsec_util -m uefi insert_before|insert_after|replace|remove <*guid*> <*rom*> <*new_rom*> <*uefi_file*>

```
chipsec_util -m uefi var-find PK
chipsec_util -m uefi var-read db D719B2CB-3D3A-4596-A3BC-DAD00E67656F db.bin
chipsec_util -m uefi var-write db D719B2CB-3D3A-4596-A3BC-DAD00E67656F db.bin
chipsec_util -m uefi var-delete db D719B2CB-3D3A-4596-A3BC-DAD00E67656F
chipsec_util -m uefi decode uefi.rom
chipsec_util -m uefi nvram uefi.rom vss_auth
chipsec_util -m uefi keys db.bin
```

ucode: provides a microcode patch command.

chipsec_util ucode id|load|decode *[ucode_update_file] [cpu_id]*
chipsec_util ucode id

```
chipsec_util ucode load ucode.bin 0
chipsec_util ucode decode ucode.pdb
```

CHIPSEC is Copyright (c) under the GNU GPLv2 general public license by Intel. This quick reference may be freely distributed under the terms of the GNU GPLv2 general public license – Copyright (c) 2017 by PreOS Security. Updates at: https://preossec.com/blog

CHIPSEC Quick Reference

for System Administrators

by PreOS Security

v0.4, 2017-07-31