„Montowanie pamięci wewnętrznej PS3 na Linuksie”

W niniejszym poradniku, dowiesz się w jaki sposób zamontować dysk twardy z konsoli PlayStation 3 na komputerze z Linuksem. Dystrybucją na której opiera się tekst jest 64 bitowy **Linux Mint** ze środowiskiem graficznym Cinnamon – nie wymaga instalacji. Wymagana jest za to konsola z wgranym dowolnym tzw. **Custom Firmware**.

I w tym miejscu e-faszystów i wszelkiej maści wojowników o czystość firmware od razu uprzedzam: hakowanie konsoli jest oczywistym łamaniem licencji, ale jeszcze nie piractwem komputerowym… Cyfrowe korsarstwo jest domeną poszukiwaczy wrażeń na kopiach gier, które nie zostały przez nich zakupione – co rzecz jasna odradzam. Poradnik kieruję do informatyków śledczych i grzebaczy-pasjonatów, którym zależy na docelowo oryginalnym firmware lub tylko tym którzy chcą mieć dostęp do danych na dysku twardym PS3 po jej tragicznej, przedwczesnej śmierci (odklejone BGA, bad bloki na NAND/NOR Flash etc.). Puryści oprogramowania i wrażliwi na informatyczny galimatias, proszeni są o opuszczenie sali (choć pewnie będą tego żałować ;)).

Jak być może wiesz, **pamięć wewnętrzna** PS3 jest zaszyfrowana unikalnym dla każdego modelu z osobna kluczami (z wyjątkiem wszystkich modeli GECR). Posiada niestandardową tablicę partycji i – zależnie od modelu – nawet dziesięć **partycji**. Przeznaczenia, ani zawartości wszystkich nie znamy, skupię się więc tylko na tych podstawowych czyli "dev_hdd0" gdzie znajdują się dane użytkownika, część systemu, pliki konfiguracyjne, plik wymiany i na "dev_flash2" gdzie znajduje się m.in. plik z ustawieniami konsoli i jej użytkowników (obecna tylko na modelach z kośćmi **NOR**).

PS3 Hard Disk Drive, partitions and storage regions

Storage Region					Access Control List			File System				Size					Usage	
OtherOS				GameOS	Secure Profile		Unk	Encryption			Type	Official				Unofficial	Official	Unofficial
					Name	auth_id		FAT		SLIM		Decimal		Hexadecimal				
								NAND	NOR	NOR		Bytes	Sectors	Bytes	Sectors			
												4 KB	8	0x1000	0x8	Same	HDD Partition Table (Physical HDD Device)	
				?	SCE_CELLOS_PME PS3_LPAR	1070000001000001 0B 1070000002000001 03		↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	Raw access	256 MB	524.288	0x10000000	0x80000	Any	First region of HDD, contains VFLASH. (only NOR)	
												4 KB	8	0x1000	0x8	Same	VFLASH Partition Table (Virtual FLASH Device, only NOR)	
			ps3vflashb (3.1(1.1))	?	SCE_CELLOS_PME	1070000001000001 03		↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	Raw access ?	14,75 MB	30.200	0xEBF000	0x75F8	Same	"mirror" of real NOR second region: "ps3nflashb" ?	
												256 KB	512	0x40000	0x200	Same	?	
			ps3vflashc (3.1(1.2))	CELL_FS_IOS:BUILTIN_FLSH1 dev_flash	PS3_LPAR SCE_CELLOS_PME PS2_*LPAR	1070000002000001 03 1070000001000001 03 1020000003000001 01		↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	FAT16	199,75 MB	409.088	0xC7C0000	0x63E00	Same	Firmware files	
			ps3vflashd (3.1(1.3))	CELL_FS_IOS:BUILTIN_FLSH2 dev_flash2	PS3_LPAR SCE_CELLOS_PME	1070000002000001 03 1070000001000001 03		↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	FAT16	16 MB	32.768	0x1000000	0x8000	Same	XRegistry (Console/User settings)	
			ps3vflashe (3.1(1.4))	CELL_FS_IOS:BUILTIN_FLSH3 dev_flash3	PS3_LPAR SCE_CELLOS_PME	1070000002000001 03 1070000001000001 03		↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	FAT12	512 KB	1.024	0x80000	0x400	Same	CRL/DRL (Bluray revocation lists)	
			ps3vflashf (3.1(1.5))	CELL_FS_IOS:BUILTIN_FLSH4 dev_flash4 ?	PS3_LPAR SCE_CELLOS_PME LINUX_LPAR	1070000002000001 03 1070000001000001 03 1080000004000001 03		↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	Raw access	4 MB	8.192	0x400000	0x2000	Any	cell_ext_os_area + OtherOS bootloader (compressed: otheros.bld)	cell_ext_os_area + OtherOS++ bootloader (uncompressed: dtbimage ps3.bin)
ps3d (3)	ps3da (3.0)											4,25 MB	8.704	0x440000	0x2200	Same	?	
			ps3vflashg (3.1(1.6))	?	SCE_CELLOS_PME	1070000001000001 03		↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	Raw access ?	256 KB	512	0x40000	0x200	Same	?	
																No ?	Padding	
			ps3vflashh (3.1(1.7))	?	No	No	No	↑:No ↓:No	↑:AES-CBC-192 (ata_key1, IV=0) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	↑:AES-XTS-128 (ata_key1, ata_key2) ↓:AES-XTS-128 (enodec_key1, enodec_key2)	All supported by linux	16,25 MB	33.280	0x1040000	0x8200	Any	Not used	Linux/FreeBSD for NOR PS3's Based on graf drivers (deprecated)
												8 KB	16	0x2000	0x10	Same ?	Padding ?	
		ps3db (3.2)		CELL_FS_UTILITY:HDD0 dev_hdd0	SCE_CELLOS_PME PS3_LPAR PS2_*LPAR	1070000001000001 03 1070000002000001 03 1020000003000001 03		AES-CBC-192 (ata_key1, IV=0)		AES-XTS-128 (ata_key1, ata_key2)	UFS2	Any					GameOS	
												4 KB	8	0x1000	0x8	Same ?	Padding	
		ps3dc (3.3)		CELL_FS_UTILITY:HDD1 dev_hdd1	SCE_CELLOS_PME PS3_LPAR	1070000001000001 03 1070000002000001 03		AES-CBC-192 (ata_key1, IV=0)		AES-XTS-128 (ata_key1, ata_key2)	FAT16	2 GB	4.194.296	0x7FFFF000	0x3FFFF8	Same	GameOS Cache	
												4 KB	8	0x1000	0x8	Same ?	Padding	
		ps3dd (3.4)		CELL_FS_UTILITY:HDD2 dev_hdd2 ?	PS3_LPAR SCE_CELLOS_PME LINUX_LPAR	1070000002000001 03 1070000001000001 03 1080000004000001 03		AES-CBC-192 (ata_key1, IV=0)		AES-XTS-128 (ata_key1, ata_key2)	All supported by linux	10GB (0x13FFFF8 sectors) All available - 10GB (All - 0x13FFFF8 sectors)				Any	OtherOS (disabled in firmware 3.21)	Linux/FreeBSD for NAND/NOR PS3's Based on glevand drivers
												4 KB	8	0x1000	0x8	Same ?	Padding	

Powyżej dla ciekawskich, wykaz logiki dysku twardego w PlayStation 3. Źródło: **psdevwiki.com**.

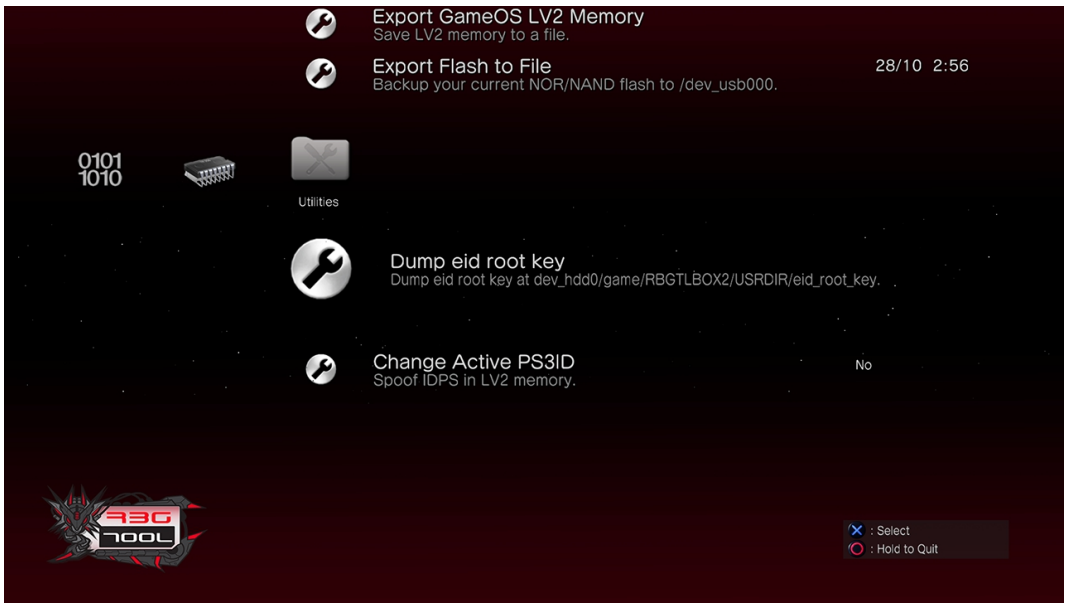
Magiczne klucze

Ze wstępu dowiedziałeś się, że są klucze **unikalne**, czyli takie które każda konsola PS3 (de facto jej płyta główna) używa do szyfrowania np. dysku twardego (to właśnie dlatego HDD z jednej konsoli nie może być odczytany na innej bez obowiązkowego formatowania, a co za tym idzie, zaszyfrowania go kluczami dla tejże konsoli).

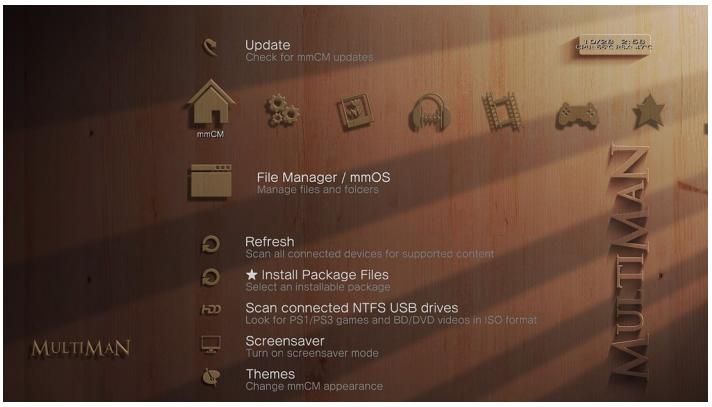
Jednym z takich kluczy jest **EID Root Key** (w dalszej części będę używać skrótu ERK), z którego za pomocą załączonego skryptu, wygenerujesz parę: **ATA Key** i **VFLASH Key**.

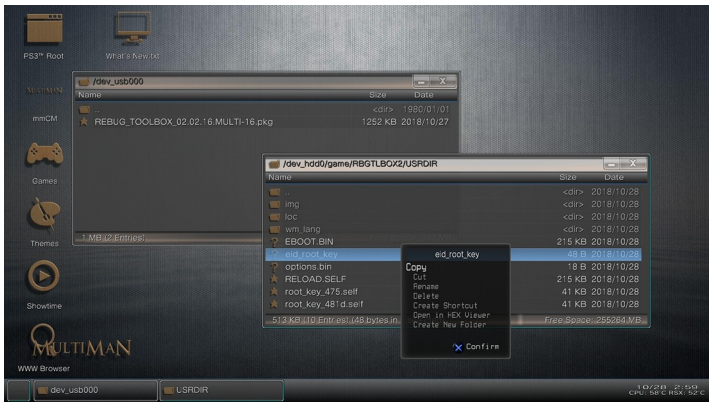
1. Na początek, na PS3 **zainstaluj** i uruchom program **Rebug Toolbox**. Oczywiście na konsoli musisz mieć wgrany CFW (oficjalny firmware, etHANol i Homebrew Enabler na to nie pozwalają) z obsługą wymaganych syscalli (polecam najnowszy Rebug REX).

2. Przejdź do kategorii "Utilities" i wybierz opcję "Dump eid root key". Konsola dwukrotnie zapiszczy i się zresetuje, a klucz wyląduje w "dev_hdd0/game/RBGTLBOX2/USRDIR/".



3. Za pomocą managera plików (np. multiMAN) skopiuj stamtąd plik "eid_root_key" na pendrive. Możesz także użyć klienta FTP jeśli masz skonfigurowane połączenie i działający serwer FTP w tle. Dodatkowo, zmień nazwę pliku na "eid_root_key**.bin**".

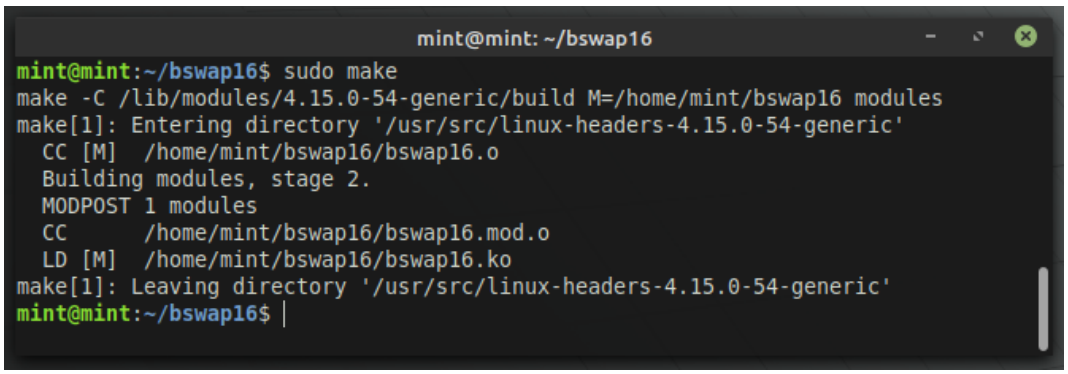


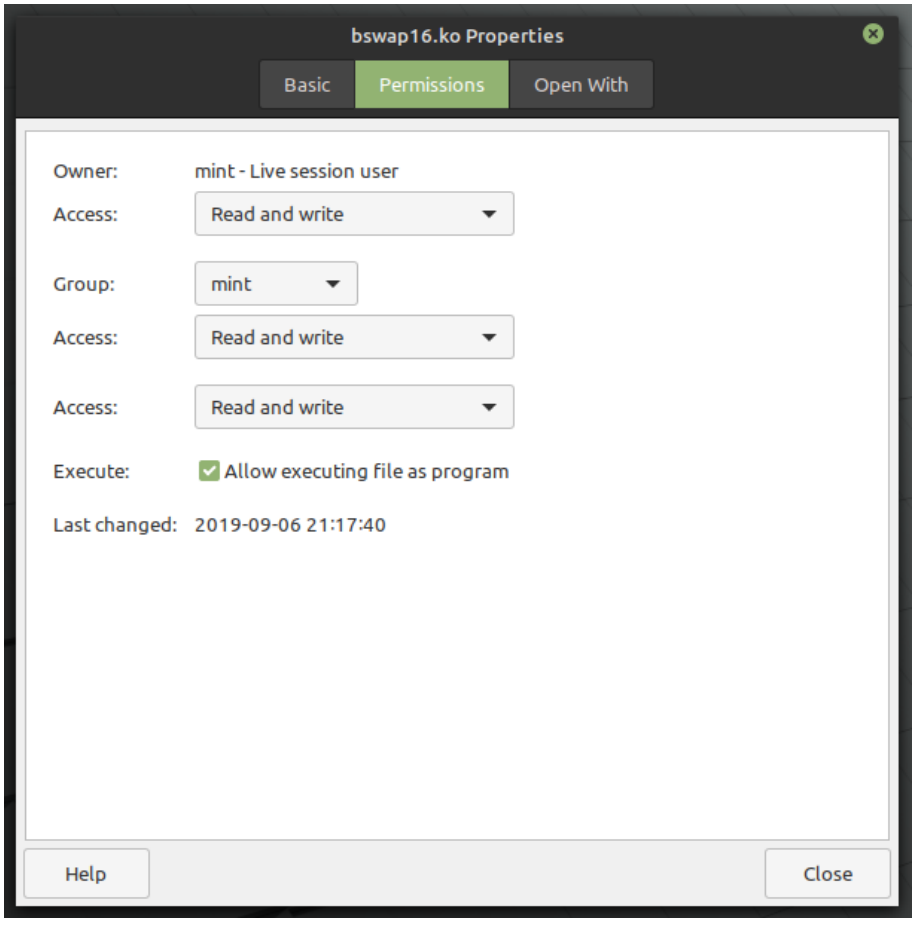


4. Wróć do komputera i stwórz folder "ps3", zaś wewnątrz niego katalogi "dev_hdd0", "dev_hdd1", "dev_hdd2", "dev_flash1", "dev_flash2" i "dev_flash3", czyli przyszłe punkty montowania partycji, które mają takie same nazwy jak w środowisku PS3.

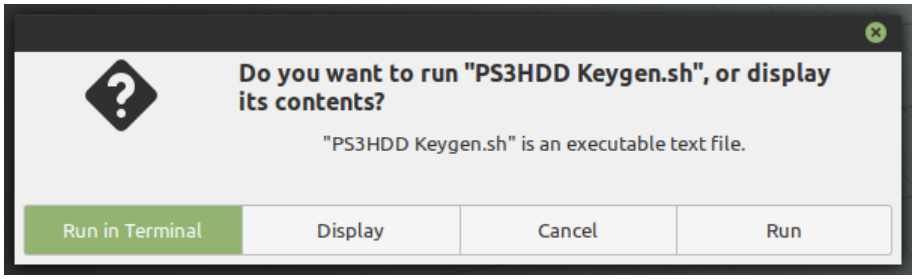
5. Pobierz i rozpakuj skrypt **PS3HDD Keygen**, zmień mu atrybuty na wykonywalny i wrzuć razem z ERK do folderu "ps3".

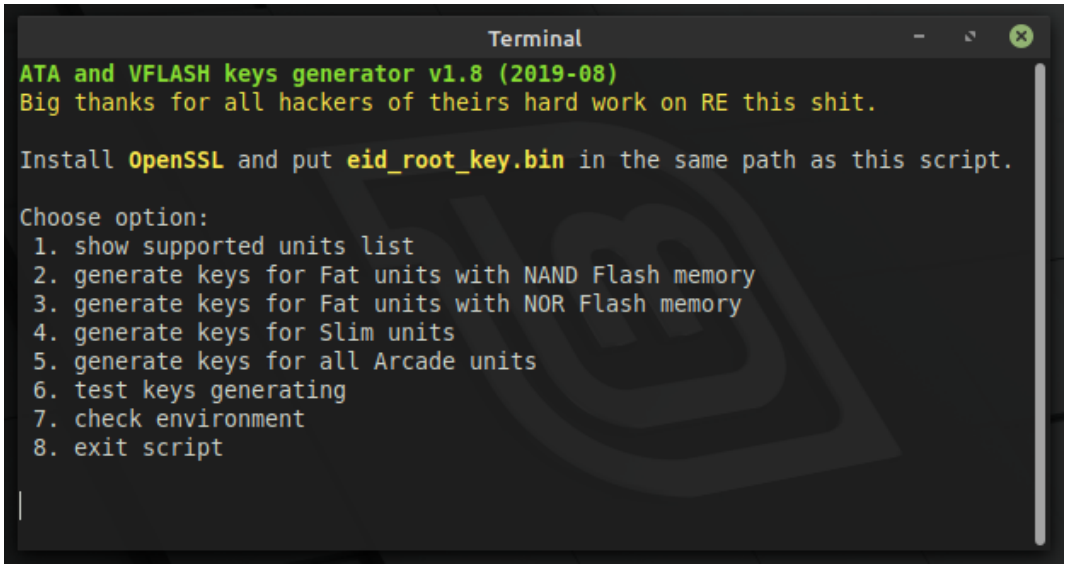
6. Pobierz moduł "bswap16.ko" lub najlepiej sam skompiluj ze źródeł, ponieważ ten konkretny będzie pasować wyłącznie do domyślnego kernela w Linux Mint 19.2 – wystarczy że otworzysz terminal, przejdziesz do katalogu z "source" i wpiszesz "sudo make"). Pamiętaj również o tym aby nadać mu atrybut wykonywalny.



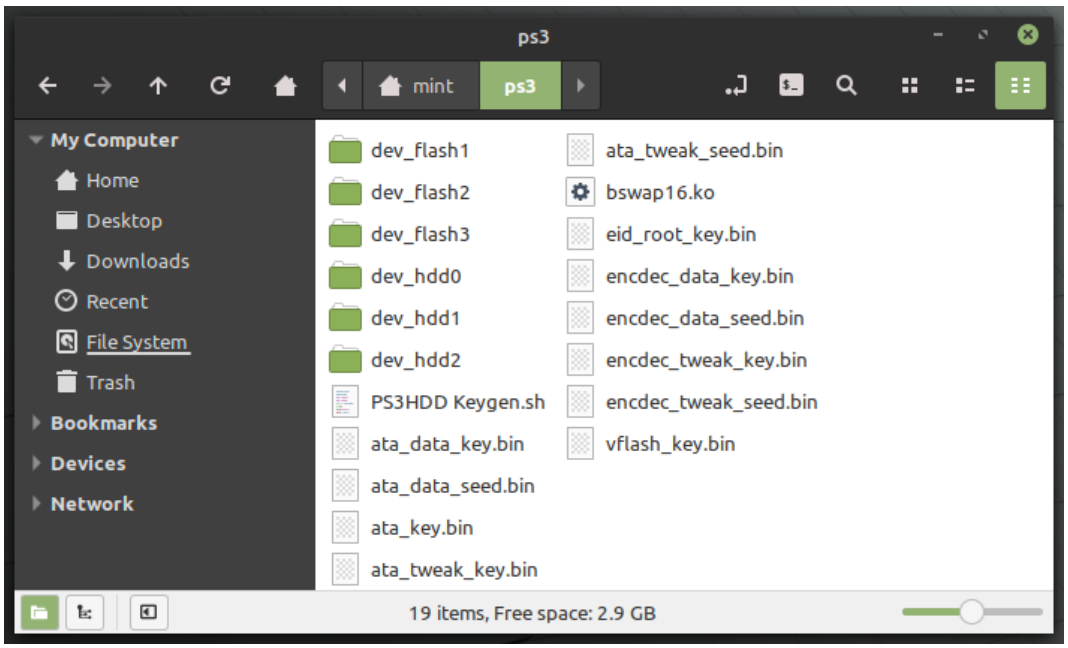


7. Uruchom generator kluczy (czyli dwu-kliknij na nim i wybierz opcję "Otwórz w terminalu") po czym wybierz model konsoli skąd pochodzi klucz i dysk twardy.





8. Po wszystkim w folderze ps3 powinny pojawić się pliki "ata_key.bin", "vflash_key.bin" plus pozostałe klucze i ziarna, które brały udział w procesie (możesz je wyrzucić, nie będą więcej potrzebne).



Deszyfracja w locie i montaż

Część najbardziej stresogenną masz dopiero przed sobą – zaledwie kilkanaście linijek do wklepania i wrota do dysku staną otworem.

- Jeśli zamierzasz podpiąć posektorowy obraz dysku twardego PS3, musisz go przypiąć jako urządzenie. W tym celu wpisz "losetup loop1 /home/mint/ps3/disk.img". Naturalnie, to tylko przykład – lokalizację i nazwę pliku musisz podać taką jaka jest u ciebie.

- Jeśli zamierzasz podpiąć prawdziwy nośnik (najlepiej bezpośrednio pod kontroler SATA, bez żadnych wynalazków po drodze typu obudowa USB) to najpierw użyj polecenia "lsblk". Dzięki temu będziesz w stanie określić nazwę urządzenia ("/dev/sda", "/dev/sdb", "/dev/sdc" itd.). W poradniku używam "/dev/sdx", ale rzec jasna to tylko przykład, więc bądź ostrożny.

1. Przejdź na prawa administratora (czyli roota) wpisując "sudo su" co potwierdzasz hasłem użytkownika. Wszystkie dalsze czynności wymagają podwyższonych uprawnień dlatego zamiast za każdym razem wklepywać sudo, wygodniej jest się na stałe przełączyć.

2. Wpisz "insmod '/home/mint/ps3/bswap16.ko'", dzięki czemu załadujesz niezbędny moduł kernela, który posłuży do stworzenia mappera, na którym w locie porządek bitowy zostanie przestawiony z Big Endian na Little Endian.

3. Następnie wpisz "cryptsetup create -c bswap16-ecb -d /dev/zero ps3hdd-bs /dev/sdx". Jeśli to obraz dysku to zastąp sdx wyrażeniem loop1.

4. Stoisz teraz przed kluczowym wyborem. Adekwatnie do rodzaju pamięci flash w konsoli, należy wybrać odpowiedni algorytm i długość klucza.

- Dla modeli z NAND Flash wpisz:
  "cryptsetup create -c aes-cbc-null -d /home/mint/ps3/ata_key.bin -s 192 ps3hdd /dev/mapper/ps3hdd-bs"

- Dla modeli z NOR Flash wpisz:
  "cryptsetup create -c aes-xts-plain64 -d /home/mint/ps3/ata_key.bin -s 256 ps3hdd /dev/mapper/ps3hdd-bs"

5. W kolejnym kroku wpisz "kpartx -a /dev/mapper/ps3hdd".

6. Sprawdź teraz za pomocą polecenia "ls -la /dev/mapper/" punkty mapowania. Powinieneś zobaczyć ps3hdd, ps3hdd1, ps3hdd2 i ps3hdd3 przekierowane na "/dev/dm-*" itp.

Na konsolach z pamięcią NOR, "ps3hdd**1**" to VFLASH, czyli wirtualny flash. Pewnego dnia, czyli wraz z **modelami** CECHHxx Sony zrezygnowało z 256 MiB pamięci NAND, na której dotychczas leżało całe oprogramowanie pokładowe (firmware + OS) na rzecz 16 MiB NOR. Resztę przeniesiono na partycję dysku twardego, którą umownie nazywamy VFLASH. Z kolei na konsolach z pamięcią NAND, mapper ten odpowiada "dev_hdd1", czyli 2GiB partycji z cache. Na wszystkich modelach, niezmienne "ps3hdd**2**" odpowiada "dev_hdd0" (czyli partycji użytkowników). Skoro na „NORówkach" "ps3hdd**1**" okupuje VFLASH to "ps3hdd**3**" pełni rolę "dev_hdd1". A jeśli masz na dysku twardym zainstalowany np. Linux (w oficjalny sposób, czyli za pomocą Other OS) to będzie jeszcze "ps3hdd**4**" odpowiadający "dev_hdd2" (na konsolach z pamięcią NAND dostanie przydział "ps3hdd**3**").

Zależnie od tego czy dysk pochodzi z konsoli z NOR czy NAND, i czy zainstalowany jest Other OS (możliwy do fw 3.15 włącznie, Other OS+ to scenowy hack dla wszystkich nowszych fw i umożliwia instalację systemu tylko na USB), lista mapperów będzie się różnić. Poradnik w dużej mierze opiera się na konsoli z NOR bez OOS i tylko dla tego tandemu możesz bezrefleksyjnie przepisywać ps3hdd*. Co jest czym, poznasz po rozmiarze (VFLASH to zawsze 256MiB, cache zawsze 2GiB, partycja użytkowników największa, a linuksowa tyle ile system sam przydzielił podczas instalacji).

7. Jeśli masz zamiar dobrać się do danych na wirtualnym flash to musisz stworzyć dodatkowe mapowanie. Ponieważ VFLASH jest szyfrowany dwukrotnie, tę czynność możesz wykonać dopiero po odszyfrowaniu i zmapowaniu „zwykłych partycji” (co zrobiłeś już wyżej).

"cryptsetup create -c aes-xts-plain64 -d /home/mint/ps3/vflash_key.bin -s 256 -p 8 ps3vflash /dev/mapper/ps3hdd1".

8. Wpisz teraz "kpartx -a /dev/mapper/ps3vflash" aby zmapować wszystkie partycje wirtualnej pamięci flash.

9. Jeśli wszystko przebiegło bez problemów, to możesz już przystąpić do montowania systemów plików (po to wcześniej tworzyłeś foldery w katalogu ps3, aby teraz zrobić z nich użytek).

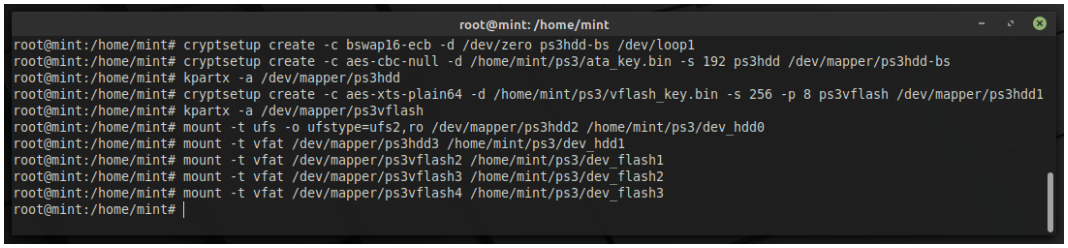
Jeśli wcześniej załadowałeś moduł kernela ufs2 umożliwiający zapis (czego poradnik nie pokrywa) to zamień "ro" na "rw".

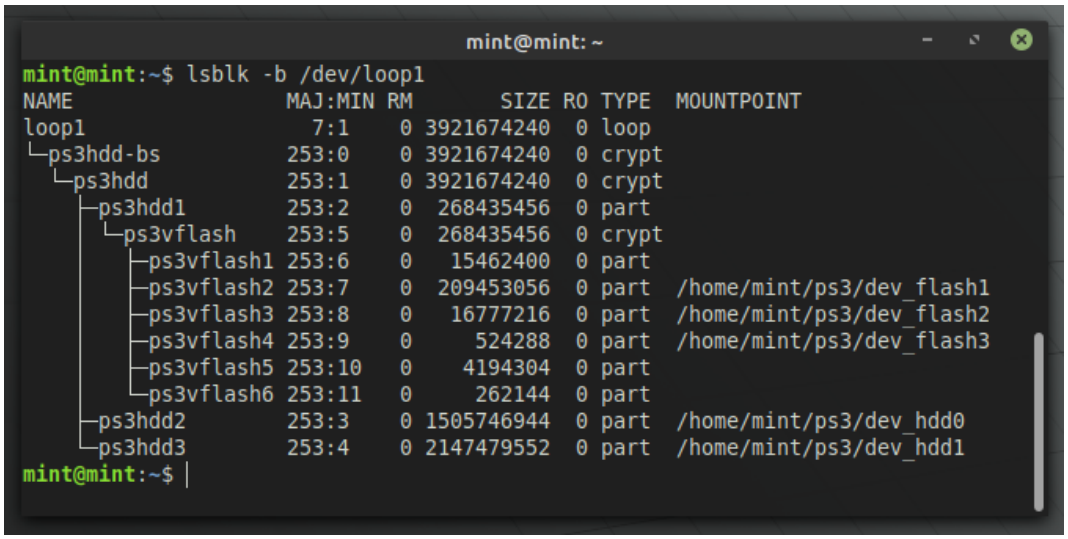
"mount -t ufs -o ufstype=ufs2,ro /dev/mapper/ps3hdd2 /home/mint/ps3/dev_hdd0"

Poniższe partycje mają systemy FAT12, FAT16 i FAT32, a więc pojawią się w managerze plików na liście do kliknięcia i automatycznego zamontowania. Tego jednak nierób ponieważ możesz mieć potem problem z demontażem (nie wiem dlaczego). Bezpieczniej jest to zrobić wklepując w terminalu:

"mount -t vfat /dev/mapper/ps3hdd3 /home/mint/ps3/dev_hdd1"
"mount -t vfat /dev/mapper/ps3vflash2 /home/mint/ps3/dev_flash1"
"mount -t vfat /dev/mapper/ps3vflash3 /home/mint/ps3/dev_flash2"
"mount -t vfat /dev/mapper/ps3vflash4 /home/mint/ps3/dev_flash3"

Uważaj na kolejność mapperów VFLASH. Może być inna niż moja. dev_flash1 zawsze będzie mieć rozmiar ~200MiB, dev_flash2 16MiB, a dev_flash3 512KiB.





W moim przypadku dyskiem twardym PS3 jest ~4GiB obraz (co jest minimalną pojemnością jaką akceptuje konsola).

10. Od tej pory uzyskałeś dostęp do wszystkich zaszyfrowanych partycji na dysku twardym PS3. Pamiętaj jednak, by nie przejmować ich na własność, jak i w ogóle nie zmieniać uprawnień. Zarządzaj danymi zawsze jako root (dla zwykłego użytkownika zawartość będzie niewidoczna). Po zamontowaniu systemów plików będziesz mógł zwiedzać zasoby managerem plików (lub w terminalu jeśli ci tak wygodniej).

Po lewej zawartość "dev_flash2/etc/", po prawej "dev_hdd0/home/<numer użytkownika konsoli>/".

Garść ciekawostek:

Na przykład dotychczas pusty folder dev_hdd0, zacznie pokazywać zawartość konsolowej partycji użytkowników. Tam jej właśnie szukaj. Jeśli przebrnąłeś przez cały poradnik, a jesteś „klikaczem Windowsa" i na Linuksa boisz się nawet spojrzeć, ;) może to być dla ciebie dezorientujące. Otóż systemy uniksowe nie montują partycji przedstawianych literami alfabetu tylko tam gdzie użytkownik chce, a więc w katalogu w jakim chce – a przecież oboje chcemy aby np. partycja użytkowników na dysku PS3 (lub jego obrazie) została zamontowana w np. "/home/<użytkownik>/ps3/dev_hdd0".

dev_hdd1 to partycja wykorzystywana głównie przez gry, ale także do aktualizacji systemu (PS3 rozpakowuje tam paczkę "**PS3UPDAT.PUP**"). W szczególnych przypadkach może się tak zdarzyć, że konsola nie będzie w stanie dokończyć aktualizacji (zatrzyma się na weryfikacji rozpakowanych danych), ani rozpocząć jej na nowo, ani tym bardziej uruchomić się w normalnym trybie. Wystarczy wtedy podłączyć dysk twardy do komputera, przejść przez opisany przeze mnie żmudny proces i skasować całą zawartość. Inaczej jedyną możliwością byłoby bezsensowne formatowanie dysku i utrata wszystkich danych… Czyli to co robią serwisy w takim przypadku lub właśnie nieszczęśnicy bez klucza EID Root Key.

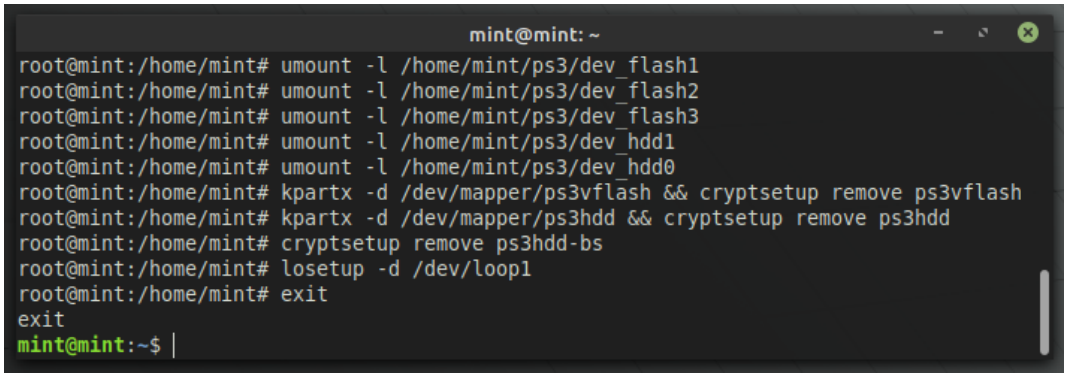
Na "dev_flash2" znajdziesz ustawienia systemu (identyfikatory kont SEN, loginy, hasła etc.) w pliku "xRegistry.sys". Warto zrobić sobie jego kopię na wypadek uszkodzenia by nie trzeba było ponownie logować się do SEN lub przepisywać save'ów, trofeów itd.

Pozostałe mappery VFLASH nie mają systemu plików, a dostęp do nich przez konsolę odbywa się najprawdopodobniej po adresach. Nie wiem co konkretnie zawierają i do czego służą (poza tymi używanymi przez OOS). Pomiędzy niektórymi znajdują się puste przestrzenie, które być może są tylko wyrównaniem, a być może mają jakąś inną rolę. Co ciekawe jest jedna partycja-lustro kości NOR (czyżby serwisowa?).

11. Kiedy już zakończysz buszowanie po strefie zakazanej czeluści dysku twardego PlayStation 3, to koniecznie musisz wszystko po kolei zdemontować. Nie powinieneś tak po prostu wyłączyć komputera, ani odpiąć dysku (a już bezwzględnie nie możesz tego zrobić jeśli zamontowałeś którąś partycję z możliwością zapisu!).

"umount -l /home/mint/ps3/dev_hdd0"
"umount -l /home/mint/ps3/dev_hdd1"
"umount -l /home/mint/ps3/dev_flash1"
"umount -l /home/mint/ps3/dev_flash2"
"umount -l /home/mint/ps3/dev_flash3"

"kpartx -d /dev/mapper/ps3vflash && cryptsetup remove ps3vflash"
"kpartx -d /dev/mapper/ps3hdd && cryptsetup remove ps3hdd"
"cryptsetup remove ps3hdd-bs"



Odpięcie loop1 rzecz jasna tylko w przypadku kiedy obraz dysku PS3 tam się znajduje.

Jeśli wszystko zrobiłeś poprawnie to konsola nigdy się nie dowie o przeprowadzonej transplantacji podczas snu. Jeśli nie, to po powrocie czeka cię wymuszone formatowanie dysku i utrata wszystkich danych. Powodzenia! :)

Najczęściej zadawane pytania

Czyli kłody jakie życie może ci rzucić pod nogi.

P: „Mam oryginalny firmware, jak odczytać ERK?”
O: Nie odczytasz. Oficjalne oprogramowanie nie pozwala na uruchamianie nieoficjalnych programów.

P: „Mam oryginalny firmware z HAN bądź HEN, jak odczytać ERK?”
O: Nie odczytasz. Ani etHANol (czyli w skrócie HAN), ani HEN (Homebrew Enabler) nie pozwalają na uruchamianie nieoficjalnych programów.

P: „Mam dump kości z firmware, jak odczytać ERK?”
O: Nie odczytasz. ERK jest częścią Meta Loadera (metldr), który jest zaszyfrowany kluczem znajdującym się w CPU, więc musiałbyś najpierw go stamtąd wydobyć, a żeby tego dokonać zhakować SYSCON (System Controller), który nim zawiaduje. Nikomu jeszcze się ta sztuka nie udała.

P: „Nie mam ERK, ale mam Drive Key lub dump Meta Loadera. Czy mogę z tego wydobyć ERK?”
O: Możesz. Drive Key, klucz używany do ożenku napędu lub przy emulacji napędu, zawiera ERK i możesz go wydłubać tym **skryptem**. W przypadku odszyfrowanego metldr (np. uzyskiwanego w procesie konwersji konsoli retail w debug za pomocą exploitu metldrpwn), ERK to jego pierwsze 48 bajtów.

Oczywiście jeśli nie masz możliwości odczytania ERK to i nie masz możliwości odczytania DK lub odszyfrowania metldr. Powyższe informacje są użyteczne dla tych, którzy kiedyś skorzystali z ODDE lub CEX2DEX, obecnie są na OFW, ale jeszcze gdzieś im się te pliki wałęsają po backupie.

P: „Podłączyłem HDD na Windows i zapytał o inicjalizację dysku, zgodziłem się”
O: Pod pojęciem „inicjalizacji dysku”, Windows rozumie nadpisanie tablicy partycji. A więc krótko: straciłeś już wszystkie dane… Ok, można to jeszcze naprawić, ale to materiał na osobny poradnik, ponieważ wymaga utworzenia posektorowego obrazu, sformatowania dysku w konsoli, sczytania tablicy, zastąpienia nią tej w obrazie, wgrania spreparowanego obrazu i czytania uważnie komunikatów. ;)

P: „Czy mogę użyć czyjegoś klucza ERK, ATA lub VFLASH?”
O: Nie możesz, są unikalne dla każdego egzemplarza konsoli. Wyjątkiem są modele GECR (czyli Arcade), ale te nie są sprzedawane detalicznie (jak nazwa sugeruje, służą za automaty w salonach gier).

P: „Zmieniłem dysk na inny, czy mogę odczytać oba, używając tego samego klucza?”
O: Tak, ponieważ klucze, którymi konsola go szyfruje nie zmieniły się…

P: „Zmieniłem konsolę i włożyłem do niej stary dysk, czy mogę go odczytać ERK ze starej konsoli?”
O: Nie możesz ponieważ jest zaszyfrowany kluczami z nowej konsoli…

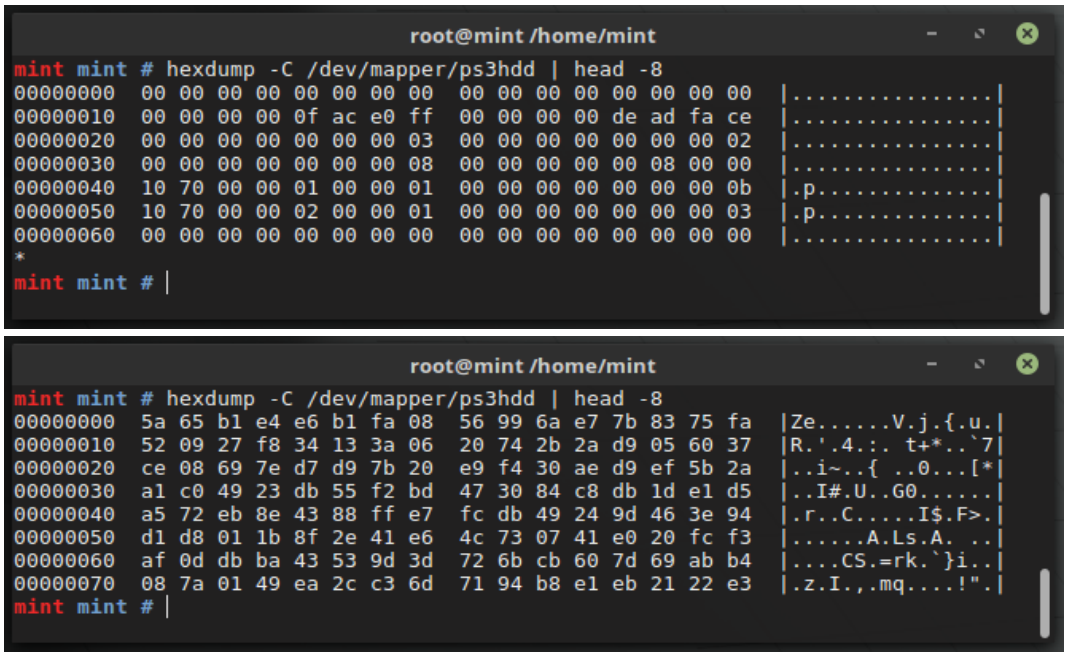
P: „Czy mogę zmienić ERK, ATA lub VFLASH w konsoli?”
O: Nie możesz.

P: „Dlaczego cryptsetup wyświetla: "device-mapper: reload ioctl on failed: No such file or directory"?”
O: Ponieważ pomyliłeś się w składni.

P: „Czy można to samo osiągnąć na FreeBSD?”
O: Nie znam się jeszcze na systemach z rodziny BSD, być może można wykorzystać narzędzia geom/geli. Niewątpliwie atutem jest domyślna i stabilna obsługa UFS2, który jest natywnym systemem plików dla FreeBSD (na którym zresztą bazuje CellOS, czyli system operacyjny PlayStation 3).

P: „Po włożeniu dysku do konsoli ta chce go formatować! Dlaczego?”
O: Dlatego, że system plików lub tablica została uszkodzona. Przyczyn może być mnóstwo, choćby taka że nie wymontowałeś partycji.

P: „Zrobiłem wszystko jak w poradniku, ale nie mam ps3hdd1, 2 i 3, dlaczego?”
O: Dlatego, że dysk nie został odszyfrowany. Prawdopodobnie niepoprawny klucz. Jeśli ERK i dysk twardy pochodzi z tej samej konsoli, to możliwe że szyfrowanie lub generowanie klucza odbywa się w inny sposób niż dla znanych nam modeli. Wklep "hexdump -C /dev/mapper/ps3hdd | head -8"), jeśli wyświetli się sieczka, a nie w większości zera, oznacza to że deszyfracja jest niepoprawna.



P: „Czy to musi być takie skomplikowane?”
O: Jeśli chcesz mieć dostęp do absolutnie wszędzie na tym dysku to musi być. Nie licząc bswap16, wykorzystujesz zwykłe narzędzia jakie można znaleźć w każdej dystrybucji Linuksa. Poradnik jest dla osób, które interesuje także zapis na tych partycjach i/lub szczegółowa analiza, jeśli chcesz tylko odczytać to użyj **HDD Readera** dla Windows lub Linux (oczywiście oba także wymagają ERK).

P: „Czy mogę uzyskać ERK na CFW, a potem wgrać oficjalny fw i w ten sposób pirackie gry?”
O: Możesz wgrać, ale nie zadziałają ponieważ materiały cyfrowe wymagają poprawnych podpisów plików wykonywalnych, poprawnie wygenerowanych i podpisanych licencji (co robi serwer Sony zanim ci je prześle), a płytowe płyt (montowanie obrazów lub katalogów wymaga programów lub wtyczek podpisanych dawno blacklistowanymi kluczami lub kluczami dla konsol debug). Podsumowując, nie, nie do tego służy dostęp do dysku na PC!

P: „A więc do czego?”
O: Jeśli zawczasu na CFW odczytasz klucze ERK i IDPS to nawet na OFW (oryginalnym oprogramowaniu) będziesz mieć pełną kontrolę nad wszystkimi danymi (włączając w to przepisywanie save’ów i trofeów pod dowolnego użytkownika, wykonywanie kopii lub przenoszenie danych na inną PS3 w razie nagłej śmierci obecnej konsoli). Bez tych kluczy jesteś skazany na łaskę i nie łaskę firmy, a nawet awaryjność swojej zabawki.

Podziękowania

- Dla **graf_chokolo** za nieoceniony wkład w inżynierię wsteczną PS3 i wsparcie Linuksa (to dzięki niemu kpartx obsługuje tablice partycji PS3).
- Dla **3141card** za HDD Reader i szereg uwag dotyczących użytych algorytmów.
- Dla **sguerrini97** za poprawienie mojego starego skryptu i przepisanie bswap16 z oryginalnego modułu jądra (niekompatybilnego z obecnymi kernelami) na program komunikujący się z nbd-client (na którym czym oparłem wcześniejszą wersję poradnika).
- Dla **anonimowego programisty** za ponowne przepisanie bswap16 i dostosowanie do obecnych kerneli.
- Dla **einsteinx2** za poradnik opisujący odblokowanie 8% wolnego miejsca na dysku twardym, a tym samym za inspirację do napisania niniejszego tekstu.
- Dla **Yugonibblit** za zrzut tablic z CECHG01 w celu potwierdzenia poprawności użytych algorytmów do wygenerowania ATA Key dla konsol z NAND.

Berion

@2018-10-28 / 2019-09-07