

Paranoid laptop

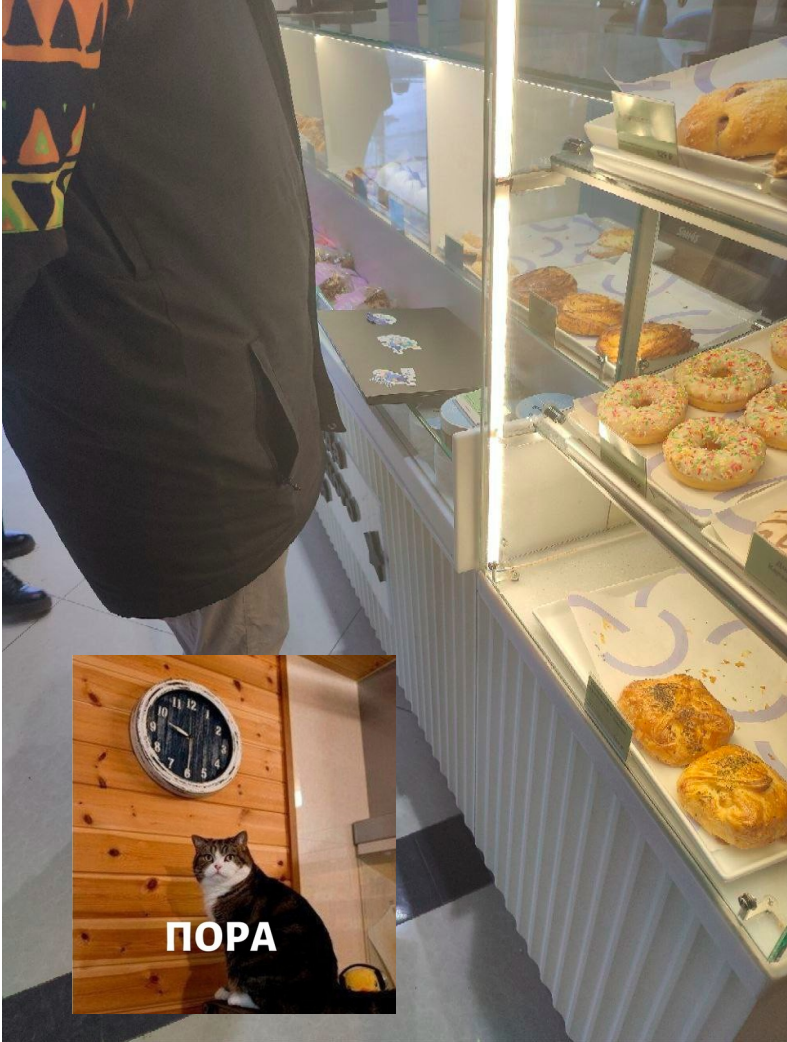
НАСТРАИВАЕМ DUAL BOOT WINDOWS+LINUX
С ПОЛНОДИСКОВЫМ ШИФРОВАНИЕМ И UEFI
SECUREBOOT БЕЗ ПЕРЕУСТАНОВКИ СИСТЕМ



ВИТАЛИЙ ЛОГИНОВ

Специалист по анализу защищенности
внешней инфраструктуры Solar JSOC

С чего все началось?



1

Диск, использующий таблицу разделов GPT

2

Наличие свободного места на разделе с Linux, равного занятому, + раздел SWAP +10%

3

Флешка с Live CD

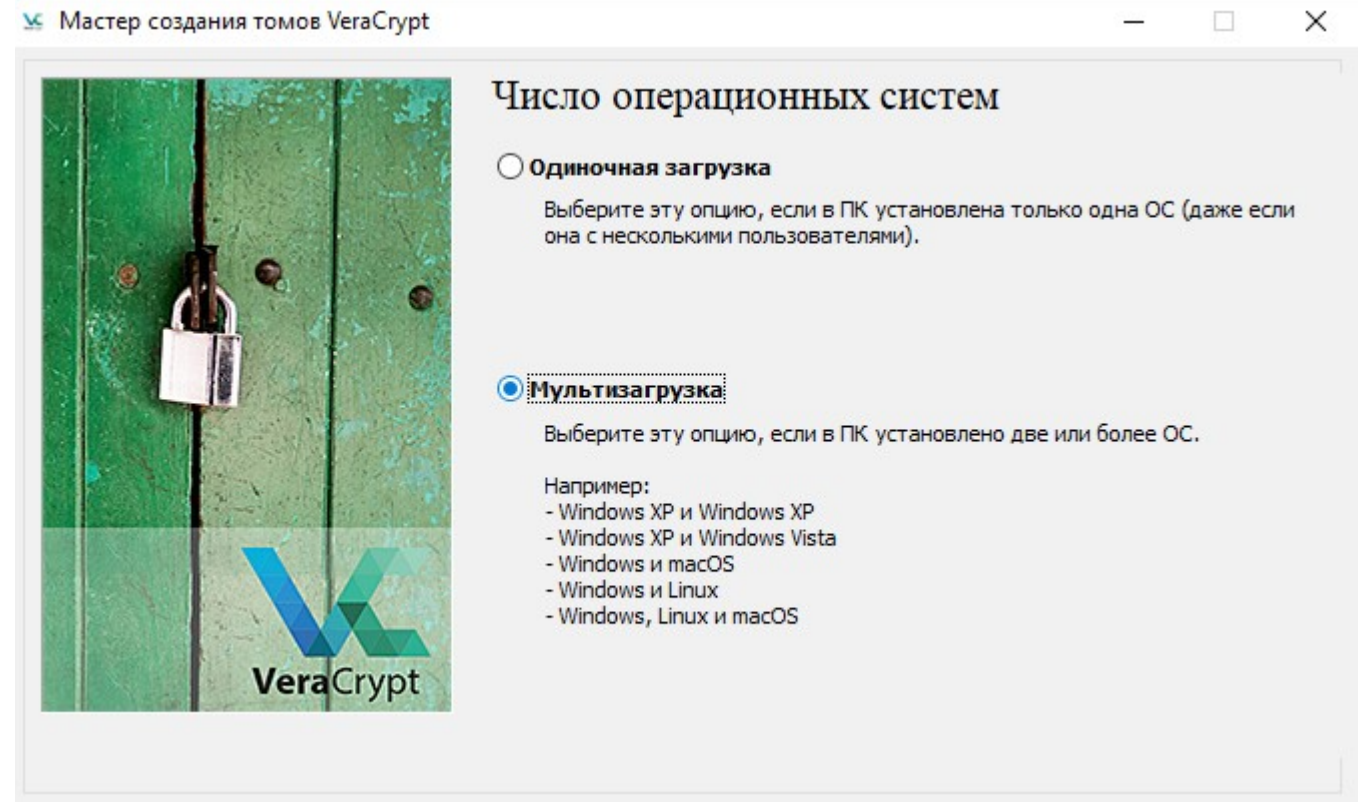
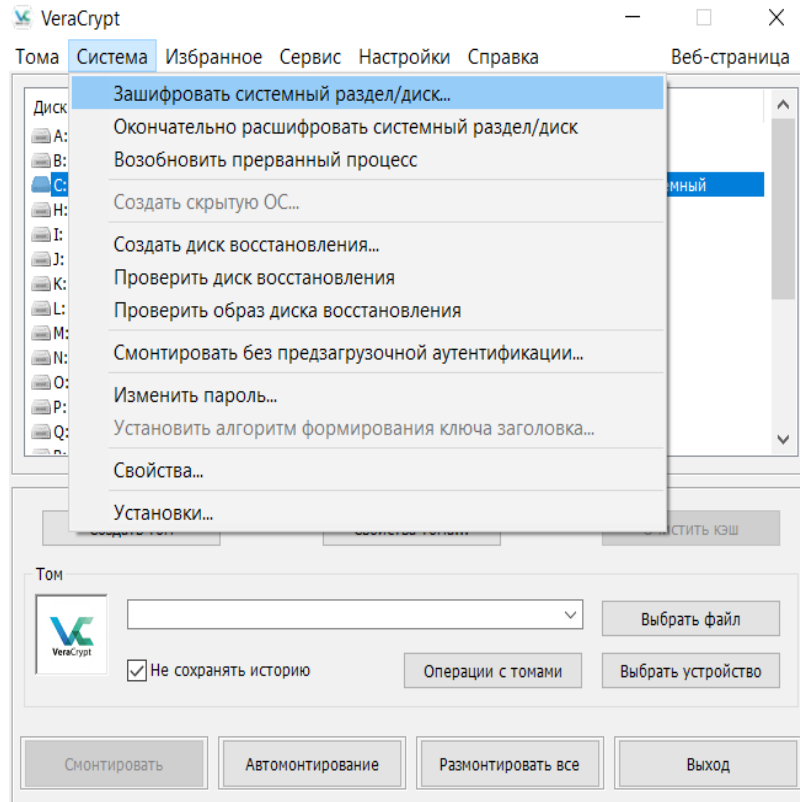
4

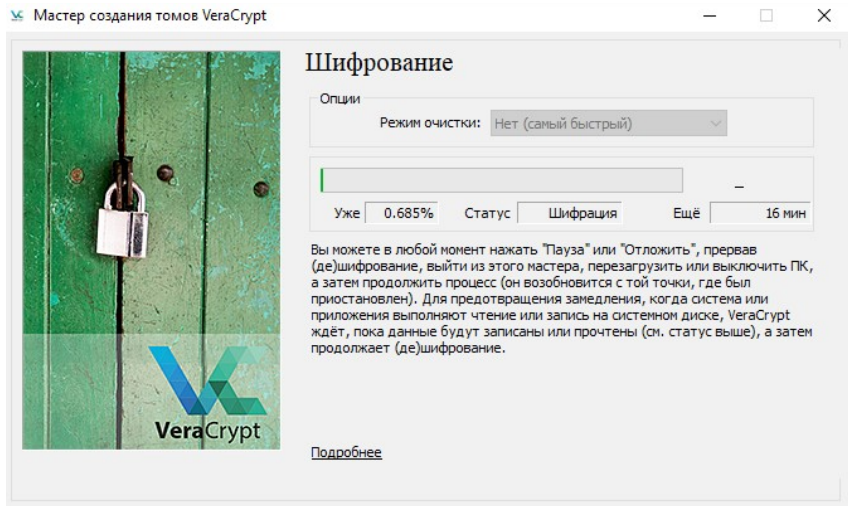
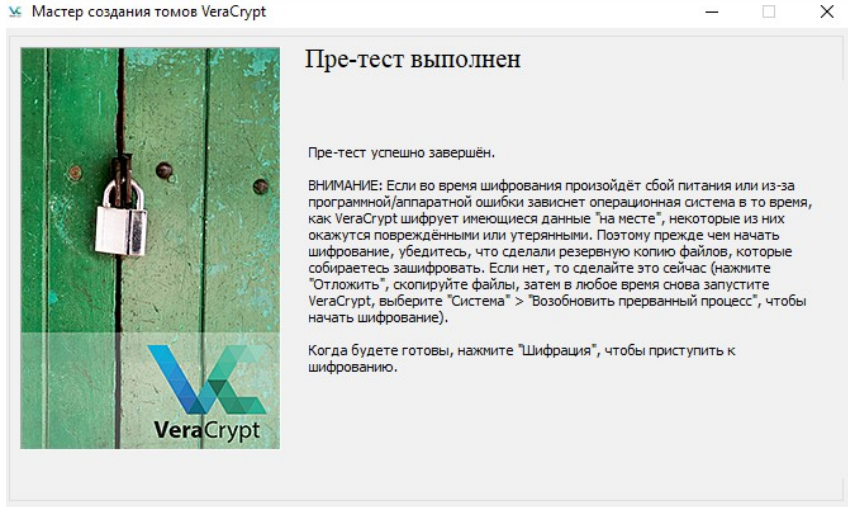
Linux (дистрибутив любой, который вам нравится, в моем случае – Kali Live CD)

РАЗДЕЛ	НАЗНАЧЕНИЕ
/dev/sda1	Служебный раздел Windows
/dev/sda2	Раздел с загрузчиками UEFI
/dev/sda3	Резервный раздел Windows
/dev/sda4	Основной системный раздел Windows, на котором расположена система
/dev/sda5	Загрузчик GRUB
/dev/sda6	Основной системный раздел Linux, на котором расположена система
/dev/sda7	Раздел файла подкачки Linux Swap

ШИФРУЕМ WINDOWS

Шифруем Windows





Шифруем Windows Pim (Персональный множитель итераций)

PIM расшифровывается как «Персональный множитель итераций».

Это параметр впервые появился в VeraCrypt 1.12, его значение определяет количество итераций, используемых функцией формирования ключа заголовка. Это значение можно указать в диалоговом окне пароля или в командной строке

ЕСЛИ ЗНАЧЕНИЕ PIM УКАЗАНО, КОЛИЧЕСТВО ИТЕРАЦИЙ ВЫЧИСЛЯЕТСЯ СЛЕДУЮЩИМ ОБРАЗОМ:

КОЛИЧЕСТВО ИТЕРАЦИЙ = PIM X 2048

Для шифрования системы без использования SHA-512 или Whirlpool

КОЛИЧЕСТВО ИТЕРАЦИЙ = 15 000 + (PIM x 1000)

Для шифрования системы с использованием SHA-512 или Whirlpool

КОЛИЧЕСТВО ИТЕРАЦИЙ = 15 000 + (PIM x 1000)

Для шифрования несистемных разделов и файлов-контейнеров

Если значение PIM не указано, будет использоваться количество итераций по умолчанию, применяемое в версиях до 1.12

ДО ВЕРСИИ 1.12 БЕЗОПАСНОСТЬ ТОМА VERACRYPT ОСНОВЫВАЛАСЬ ТОЛЬКО НА НАДЕЖНОСТИ ПАРОЛЯ, ПОСКОЛЬКУ VERACRYPT ИСПОЛЬЗОВАЛ ФИКСИРОВАННОЕ КОЛИЧЕСТВО ИТЕРАЦИЙ

Благодаря реализации управления PIM у VeraCrypt, появилось двумерное пространство безопасности для томов, основанное на паре (Пароль, PIM). Это обеспечивает большую гибкость при настройке желаемого уровня безопасности, одновременно контролируя производительность операции монтирования/загрузки

ШИФРУЕМ LINUX

Создание раздела для переноса ОС

● ЗАГРУЖАЕМСЯ С LIVE CD

ЗАПУСКАЕМ GPARTED

ОТРЕЗАЕМ СВОБОДНОЕ МЕСТО ОТ РАЗДЕЛА
С LINUX И ПОЛУЧАЕМ НЕРАЗМЕЧЕННУЮ ОБЛАСТЬ

В ПОЛУЧЕННОЙ ОБЛАСТИ
СОЗДАЕМ НОВЫЙ РАЗДЕЛ

INPUT = {PARANOID_AND.COMPLICATED}

```
cryptsetup luksFormat --cipher aes-xts-plain64 --key-size 512 --hash sha512 --iter-time 2000 /dev/sda8
```

ОПЦИИ

[01]

luksFormat – инициализация
LUKS-заголовка

[02]

--key-size – длина ключа
для алгоритма шифрования

[03]

/dev/sda8 – раздел для переноса ОС,
созданный в предыдущем пункте

[04]

--cipher – выбранный
алгоритм шифрования

[05]

--hash – выбранный алгоритм
хеширования

[06]

--iter-time – время в миллисекундах,
затрачиваемое на генерацию ключа
из вводимого пароля функцией PBKDF2

```
cryptsetup open /dev/sda8 sda8_crypt
```

ОПЦИИ

[01]

open – сопоставить раздел «с именем»

[02]

/dev/sda8 – имя созданного ранее раздела

[03]

sda8_crypt – выбранное имя криптоконтейнера, которое используется для монтирования зашифрованного раздела или его инициализации при загрузке ОС

РЕЗУЛЬТАТ

Контейнер доступен как блочное устройство
/dev/mapper/sda8_crypt

Создание структуры LVM на зашифрованном разделе

[01]

```
pvcreate /dev/mapper/sda8_crypt
```

[03]

```
lvcreate -n swap -L 4G kali
```

[05]

```
lvcreate -n root -L 70G kali
```

[02]

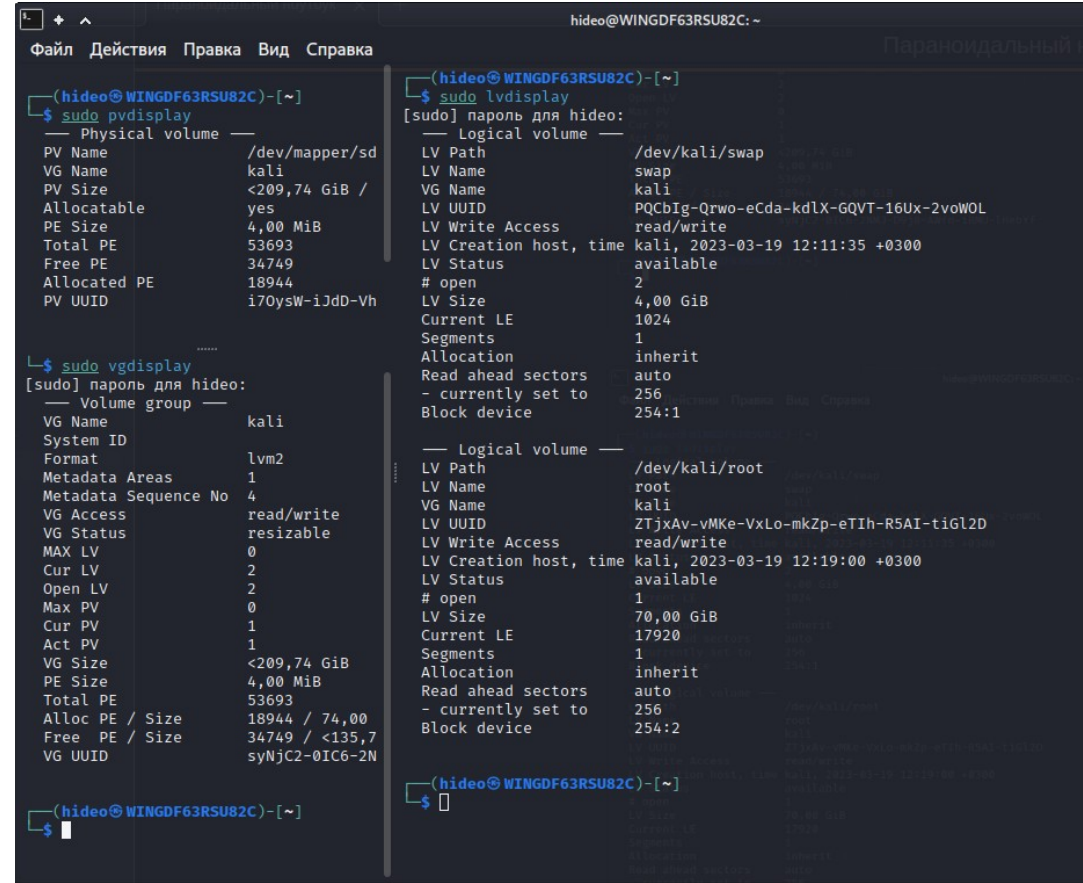
```
vgcreate kali /dev/mapper/sda8_crypt
```

[04]

```
mkswap /dev/ubuntu/swap
```

[06]

```
mkfs.ext4 /dev/kali/root
```



```
hideo@WINGDF63RSU82C: ~  
Файл Действия Правка Вид Справка  
(hideo@WINGDF63RSU82C)-[~]  
└─$ sudo pvdisplay  
Physical volume  
PV Name /dev/mapper/sd  
VG Name kali  
PV Size <209,74 GiB /  
Allocatable yes  
PE Size 4,00 MiB  
Total PE 53693  
Free PE 34749  
Allocated PE 18944  
PV UUID i70ysW-iJdD-Vh  
  
└─$ sudo vgdisplay  
[sudo] пароль для hideo:  
Volume group  
VG Name kali  
System ID  
Format lvm2  
Metadata Areas 1  
Metadata Sequence No 4  
VG Access read/write  
VG Status resizable  
MAX LV 0  
Cur LV 2  
Open LV 2  
Max PV 0  
Cur PV 1  
Act PV 1  
VG Size <209,74 GiB  
PE Size 4,00 MiB  
Total PE 53693  
Alloc PE / Size 18944 / 74,00  
Free PE / Size 34749 / <135,7  
VG UUID syNjC2-0IC6-2N  
  
(hideo@WINGDF63RSU82C)-[~]  
└─$ sudo lvdisplay  
[sudo] пароль для hideo:  
Logical volume  
LV Path /dev/kali/swap  
LV Name swap  
VG Name kali  
LV UUID PQCbIg-Qrwo-eCda-kdLX-GQVT-16Ux-2voWOL  
LV Write Access read/write  
LV Creation host, time kali, 2023-03-19 12:11:35 +0300  
LV Status available  
# open 2  
LV Size 4,00 GiB  
Current LE 1024  
Segments 1  
Allocation inherit  
Read ahead sectors auto  
- currently set to 256  
Block device 254:1  
  
Logical volume  
LV Path /dev/kali/root  
LV Name root  
VG Name kali  
LV UUID ZTjxAv-vMKe-VxLo-mkZp-eTIH-R5AI-tiGL2D  
LV Write Access read/write  
LV Creation host, time kali, 2023-03-19 12:19:00 +0300  
LV Status available  
# open 1  
LV Size 70,00 GiB  
Current LE 17920  
Segments 1  
Allocation inherit  
Read ahead sectors auto  
- currently set to 256  
Block device 254:2  
  
(hideo@WINGDF63RSU82C)-[~]  
└─$
```

```
mount /dev/mapper/kali-root /mnt  
mount /dev/sda6 /mnt2  
rsync -avlxhHX --progress /mnt2/ /mnt
```

ОПЦИИ

[01]

-a – режим архива

[02]

-v – вербализация

[03]

-l – копировать символичные ссылки

[04]

-x – работать только в этой файловой системе

[05]

-H – копировать, хардлинки как есть

[06]

-P – progress – статус времени работы над файлом

Настройка системы на зашифрованном разделе

СОЗДАЕМ ФАЙЛЫ-МАРКЕРЫ
НА ШИФРОВАННОЙ И НЕШИФРОВАННОЙ ОС

```
drwxr-xr-x  8 root root 4,0K фев 12 19:50 opt
-rw-r--r--  1 root root  0 map 19 17:39 os_decrypt
drwxr-xr-x  2 root root 4,0K map 19 11:23 proc
drwx----- 8 root root 4,0K map 21 20:25 root
```

```
drwxr-xr-x  2 root root 4,0K map 20 23:41 mnt2
drwxr-xr-x  8 root root 4,0K фев 12 19:50 opt
-rw-r--r--  1 root root  0 map 19 17:40 os_encrypt
dr-xr-xr-x 371 root root  0 map 22 16:11 proc
drwx-----  8 root root 4,0K map 21 20:25 root
drwxr-xr-x 36 root root 920 map 22 16:30 run
```

Настройка системы на зашифрованном разделе

[01] СВЯЗЫВАЕМ

[/dev, /sys /proc] ————— [/mnt/dev, /mnt/sys /mnt/proc]

а также

/etc/resolv.conf ————— /mnt/etc/resolv.conf

чтобы у вас был доступ к сети

```
mount --bind /dev /mnt/dev
```

```
mount --bind /sys /mnt/sys
```

```
mount --bind /proc /mnt/proc
```

```
mount --bind /etc/resolv.conf /mnt/etc/resolv.conf
```

```
chroot /mnt
```

```
Mount /dev/sda2 /boot/efi
```

[02] МЕНЯЕМ КОРНЕВОЙ КАТАЛОГ С ПОМОЩЬЮ CHROOT

[03] МОНТИРУЕМ ESP-РАЗДЕЛ С ЗАГРУЗЧИКАМИ

Настройка системы на зашифрованном разделе. Настройка crypttab и fstab

● ЗАПУСКАЕМ ОБЗОР БЛОЧНЫХ УСТРОЙСТВ:

SUDO BLKID

НАХОДИМ СТРОКУ, НАЧИНАЮЩУЮСЯ
С /DEV/SDA8, ТАМ БУДЕТ ЗАПИСАН UUID

КОПИРУЕМ ЕГО

ОТКРЫВАЕМ CRYPTTAB:

SUDO NANO /ETC/CRYPTTAB

ДОБАВЛЯЕМ ЗАПИСЬ ВИДА

SDA8_CRYPT UUID={СКОПИРОВАННОЕ
ЗНАЧЕНИЕ} NONE LUKS

СОХРАНЯЕМ ФАЙЛ

```
GNU nano 7.2
# <target name> <source device>          <key file>          <options>
sda8_crypt UUID=f3fba16f-6c8b-4c5b-8ab8-20da1ddb5fd9 none luks
```

В /ETC/FSTAB ВНОСИМ НАШИ ТОМА LVM:

```
GNU nano 7.2
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# systemd generates mount units based on this file, see systemd.mount(5).
# Please run 'systemctl daemon-reload' after making changes here.
#
# <file system> <mount point>  <type> <options>          <dump> <pass>
# / was on /dev/sda6 during installation
# Bootability restore: Change id = C8C1C25C-5E1A-47B5-ACE0-7F0B8E54F33F
/dev/mapper/kali-root / ext4 errors=remount-ro 0 1
# swap was on /dev/sda7 during installation
/dev/mapper/kali-swap none swap sw          0 0
```


ПЕРЕСОБИРАЕМ INITRAMFS

```
cp /boot/initrd.img-6.1.0-kali5-amd64{,.backup1}
```

```
echo -n "quite splash" > /tmp/cmdline  
make-initrd
```



ОБРАТИТЕ ВНИМАНИЕ:

Для работы с lvm и шифрованием нужны специальные модули ядра. make-initrd использует встроенный модуль guess для создания оптимальной конфигурации. С другими утилитами не все так очевидно, в частности, mkinitcpio, используемый в Arch Linux, требует указания конкретных хуков для добавления нужных модулей.

```
root@WINGDF63RSU82C: /home/hideo/tools/make-initrd
Файл Действия Правка Вид Справка

(root@WINGDF63RSU82C)-[/home/hideo/tools/make-initrd]
# !386

(root@WINGDF63RSU82C)-[/home/hideo/tools/make-initrd]
# make-initrd
[00:00:00] Config file: /etc/initrd.mk
[00:00:06] Generating module dependencies on host ...
[00:00:25] Used features: add-modules add-udev-rules cleanup compress depmod-image devmapper luks lvm modules-crypt
o-user-api network rdshell rootfs system-glibc ucode
[00:00:25] Packed modules: af_alg ahci algif_aead algif_hash algif_rng algif_skcipher authenc crc16 crc32c_generic
crc32c-intel crc64 crc64-rocksoft crc-t10dif crct10dif_common crct10dif_generic crct10dif_pclmul dm-bufio dm-crypt
dm-mod dm-snapshot ecb essiv evdev ext4 hid hid-generic jbd2 libahci libata mbcache scsi_common scsi_mod sd_mod ser
io_raw t10-pi xts
[00:00:25] Unpacked size: 46M
[00:00:25] Image size: 13M
[00:00:25] Image is saved as /boot/initrd.img-6.1.0-kali5-amd64
```

СПИСОК НУЖНЫХ ХУКОВ ДЛЯ ARCH:

- | | | |
|-----------------|------------------|------------------|
| [01] base | [06] keyboard | [11] lvm2 |
| [02] udev | [07] keymap | [12] filesystems |
| [03] autodetect | [08] consolefont | [13] fsck |
| [04] modconf | [09] block | |
| [05] kms | [10] encrypt | |

СОЗДАЕМ UNIFIED KERNEL IMAGE

ВЫЧИСЛЯЕМ ПОБАЙТОВЫЙ СДВИГ ДЛЯ КОМПОНЕНТОВ УНИВЕРСАЛЬНОГО ОБРАЗА:

- Сведения о релизе системы
- Аргументы командной строки параметров ядра
- Образ Initramfs
- Ядро системы

ВСТРАИВАЕМ КОМПОНЕНТЫ В ЗАГОТОВКУ .efi ПРИЛОЖЕНИЯ И ПОЛУЧАЕМ ЗАГРУЗЧИК СИСТЕМЫ

```
$ align="$(objdump -p /usr/lib/systemd/boot/efi/linuxx64.efi.stub | awk
'{ if ($1 == "SectionAlignment"){print $2} }')"
$ align~~=$((16#$align))
$ osrel_offs="$(objdump -h "/usr/lib/systemd/boot/efi/linuxx64.efi.stub"
| awk 'NF==7 {size=strtonum("0x"$3); offset=strtonum("0x"$4)} END {print
size + offset}')"
$ osrel_offs=$((osrel_offs + "$align" - osrel_offs % "$align"))
$ cmdline_offs=$((osrel_offs + $(stat -Lc%s "/usr/lib/os-release")))
$ cmdline_offs=$((cmdline_offs + "$align" - cmdline_offs % "$align"))
$ initrd_offs=$((cmdline_offs + $(stat -Lc%s "/tmp/cmdline")))
$ initrd_offs=$((initrd_offs + "$align" - initrd_offs % "$align"))
$ linux_offs=$((initrd_offs + $(stat -Lc%s "/boot/initrd.img-6.1.0-kali5-
amd64")))
$ linux_offs=$((linux_offs + "$align" - linux_offs % "$align"))
$ objcopy \
  --add-section .osrel="/usr/lib/os-release" --change-section-vma
.osrel=$(printf 0x%x $osrel_offs) \
  --add-section .cmdline="/tmp/cmdline" \
  --change-section-vma .cmdline=$(printf 0x%x $cmdline_offs) \
  --add-section .initrd="/boot/initrd.img-6.1.0-kali5-amd64" \
  --change-section-vma .initrd=$(printf 0x%x $initrd_offs) \
  --add-section .linux="/boot/vmlinuz-6.1.0-kali5-amd64~~" \
  --change-section-vma .linux=$(printf 0x%x $linux_offs) \
  "/usr/lib/systemd/boot/efi/linuxx64.efi.stub"
"/boot/efi/EFI/crypt_kali/crypt_kali.efi"
```

Итоговая структура разделов

```
root@WINGDF63RSU82C: ~  
Файл Действия Правка Вид Справка  
  
(root@WINGDF63RSU82C)-[~]  
# lsblk  
NAME                MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS  
sda                  8:0    0 931,5G  0 disk  
├─sda1                8:1    0   529M  0 part  
├─sda2                8:2    0    99M  0 part /boot/efi  
├─sda3                8:3    0    16M  0 part  
├─sda4                8:4    0 651,6G  0 part  
└─┬─veracrypt1       254:3   0 651,6G  0 dm   /media/hideo/windows_10  
   └─sda8             8:8    0 209,8G  0 part  
     └─sda8_crypt     254:0   0 209,7G  0 crypt  
        ├──kali-swap  254:1   0    4G   0 lvm   [SWAP]  
        └─kali-root   254:2   0   70G   0 lvm   /  
sdb                  8:16   1    0B   0 disk  
  
(root@WINGDF63RSU82C)-[~]  
#
```

НАВОДИМ КРАСОТУ

Установка и настройка rEFInd

УСТАНАВЛИВАЕМ НЕОБХОДИМЫЕ ПАКЕТЫ

```
pacman -S refind  
sudo pacman -S efibootmgr
```

МОНТИРУЕМ ЗАГРУЗОЧНЫЙ РАЗДЕЛ

```
mount /dev/sda2 /boot/efi  
mkdir -p /boot/efi/EFI/refind  
cp /usr/share/refind/refind.conf-sample /boot/efi/EFI/refind/refind.conf
```

```
hideo@archlinux /mnt$ sudo pacman -S refind  
[sudo] password for hideo:  
warning: refind-0.14.0.2-1 is up to date -- reinstalling  
resolving dependencies...  
looking for conflicting packages...  
  
Packages (1) refind-0.14.0.2-1  
  
Total Installed Size: 1.92 MiB  
Net Upgrade Size: 0.00 MiB  
  
:: Proceed with installation? [Y/n] y  
(1/1) checking keys in keyring  
(1/1) checking package integrity  
(1/1) loading package files  
(1/1) checking for file conflicts  
(1/1) checking available disk space  
:: Processing package changes...  
(1/1) reinstalling refind  
:: Running post-transaction hooks...  
(1/1) Arming ConditionNeedsUpdate...
```

НАСТРАИВАЕМ ТОЧКУ ЗАГРУЗКИ

```
efibootmgr --create --disk /dev/sda --part 2 --loader /EFI/refind/refind.efi --label "rEFInd Boot Manager" --unicode
```

ОПЦИИ

[01]

--create – создать новую точку загрузки и добавить в список загрузок

[02]

--disk – диск, на котором расположен загрузчик

[03]

--part – раздел диска, на котором расположен загрузчик

[04]

--loader – сам загрузчик

[05]

--label – отображаемое имя

[06]

--unicode – кодировка параметров командной строки загрузки ядра

```
hideo@archlinux /boot/efi/EFI/refind$ sudo efibootmgr --create --disk /dev/sda --part 2 --loader /EFI/ref
[sudo] password for hideo:
BootCurrent: 0000
Timeout: 2 seconds
BootOrder: 0002,0000,0001,0003,0019,001A,001B,0017,0018,001C,001D,001E,0023,DC5B
Boot0000* Refind          HD(2,GPT,a0be39bb-8f3f-402b-9c54-7d11bbe2be69,0x109000,0x31800)/File(\EFI\refind\
Boot0001* kali           HD(2,GPT,a0be39bb-8f3f-402b-9c54-7d11bbe2be69,0x109000,0x31800)/File(\EFI\kali\grubx64.ef
Boot0003* Windows Boot Manager HD(2,GPT,a0be39bb-8f3f-402b-9c54-7d11bbe2be69,0x109000,0x31800)/File(\EFI
Boot0010 Setup           FvFile(721c8b66-426c-4e86-8e99-3457c46ab0b9)
Boot0011 Boot Menu       FvFile(126a762d-5758-4fca-8531-201a7f57f850)
Boot0012 Diagnostic Splash Screen FvFile(a7d8d9a6-6ab0-4aeb-ad9d-163e59a7a380)
Boot0013 Lenovo Diagnostics FvFile(3f7e615b-0d45-4f80-88dc-26b234958560)
Boot0014 Startup Interrupt Menu FvFile(f46ee6f4-4785-43a3-923d-7f786c3c8479)
Boot0015 Rescue and Recovery FvFile(665d3f60-ad3e-4cad-8e26-db46eee9f1b5)
Boot0016 MEBx Hot Key FvFile(ac6fd56a-3d41-4efd-alb9-870293811a28)
Boot0017 USB CD          VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,86701296aa5a7848b66cd49dd3ba6a55)
Boot0018 USB FDD         VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,6ff015a28830b543a8b8641009461e49)
Boot0019* NVMe0          VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,001c199932d94c4eae9aa0b6e98eb8a400)
Boot001A* ATA HDD0       VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,91af625956449f41a7b91f4f892ab0f600)
Boot001B* USB HDD        VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,33e821aaaf33bc4789bd419f88c50803)
Boot001C PCI LAN         VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,78a84aaf2b2afc4ea79cf5cc8f3d3803)
Boot001D Other CD        VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,aea2090adfde214e8b3a5e471856a35406)
Boot001E Other HDD       VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,91af625956449f41a7b91f4f892ab0f606)
Boot001F* USBR BOOT CDROM PciRoot(0x0)/Pci(0x14,0x0)/USB(11,1)
Boot0020* USBR BOOT Floppy PciRoot(0x0)/Pci(0x14,0x0)/USB(11,0)
Boot0021* ATA HDD        VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,91af625956449f41a7b91f4f892ab0f6)
Boot0022* ATAPI CD       VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,aea2090adfde214e8b3a5e471856a354)
Boot0023* PCI LAN         VenMsg(bc7838d2-0f82-4d60-8316-c068ee79d25b,78a84aaf2b2afc4ea79cf5cc8f3d3803)
BootDC5B* VeraCrypt BootLoader (DcsBoot) HD(2,GPT,a0be39bb-8f3f-402b-9c54-7d11bbe2be69,0x109000,0x
Boot0002* rEFInd Boot Manager HD(2,GPT,a0be39bb-8f3f-402b-9c54-7d11bbe2be69,0x109000,0x31800)/File(\EFI
```

Установка и настройка rEFInd

НАСТРАИВАЕМ КОНФИГУРАЦИЮ

```
# Ожидание в секундах перед авто-выбором ОС
timeout 20
use_nvram false
# фоновый рисунок
banner /EFI/refind/m.png
# отключить автоматическое обнаружение загрузчиков в директориях
dont_scan_dirs /EFI/VeraCrypt
dont_scan_dirs /EFI/Microsoft

# Пункт для загрузки windows
menuentry Windows {
    icon \EFI\refind\icons\blue.png
    loader \EFI\Boot\bootx64.efi
}

# Пункт для загрузки Arch
menuentry Arch {
    icon /EFI/refind/icons/os_arch.png
    loader /EFI/arch/arch.efi
}

# Пункт для загрузки Kali
menuentry kali {
    icon /EFI/refind/icons/red.png
    loader /EFI/crypt_kali/crypt_kali.efi
}
```

ЭКРАН ЗАГРУЗКИ ПОСЛЕ УСТАНОВКИ

```
rEFInd - Main Menu

Boot EFI\refind_backup\refind_x64.efi from EFI system partition
Boot EFI\arch\arch.efi from EFI system partition
Boot EFI\arch\arch_old.efi from EFI system partition
Boot EFI\VeraCrypt\DcsBoot.efi from EFI system partition
Boot Fallback boot loader from EFI system partition
About rEFInd
Manage Hidden Tags Menu
Shut Down Computer
Reboot Computer
Reboot to Computer Setup Utility
```

СОДЕРЖИМОЕ ПАПКИ

```
hideo@archlinux /boot/efi/EFI/refind$ ls -la
total 750K
drwxr-xr-x 4 root root 1.0K Oct 12 11:33 .
drwxr-xr-x 8 root root 1.0K Oct 12 11:04 ..
drwxr-xr-x 3 root root 7.0K Oct 12 11:33 icons
-rwxr-xr-x 1 root root 415K Oct 12 11:33 m.png
-rwxr-xr-x 1 root root 301 Oct 12 11:33 refind.conf
-rwxr-xr-x 1 root root 324K Oct 11 23:26 refind.efi
drwxr-xr-x 2 root root 1.0K Oct 12 11:04 vars
```

Установка и настройка rEFInd. Финальный вид



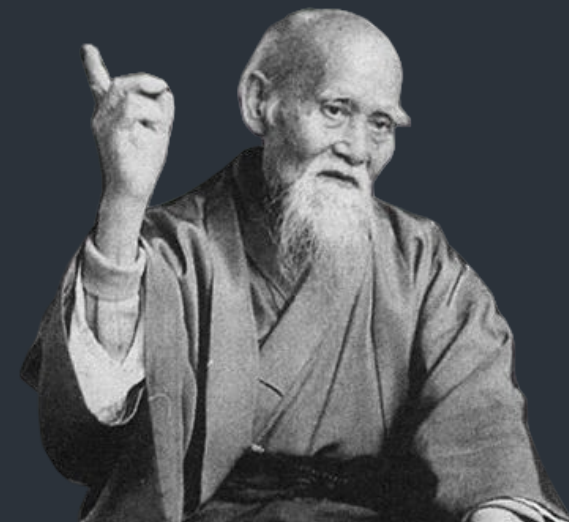
UEFI SECURE BOOT

ОТКАЗ ОТ ОТВЕТСТВЕННОСТИ

Все следующие действия выполняются на свой страх и риск и могут привести к полной неработоспособности ПК!

Админы делятся на тех, кто делает бэкапы,
и на тех, кто ТЕПЕРЬ делает бэкапы

(с) Старая айтишная мудрость

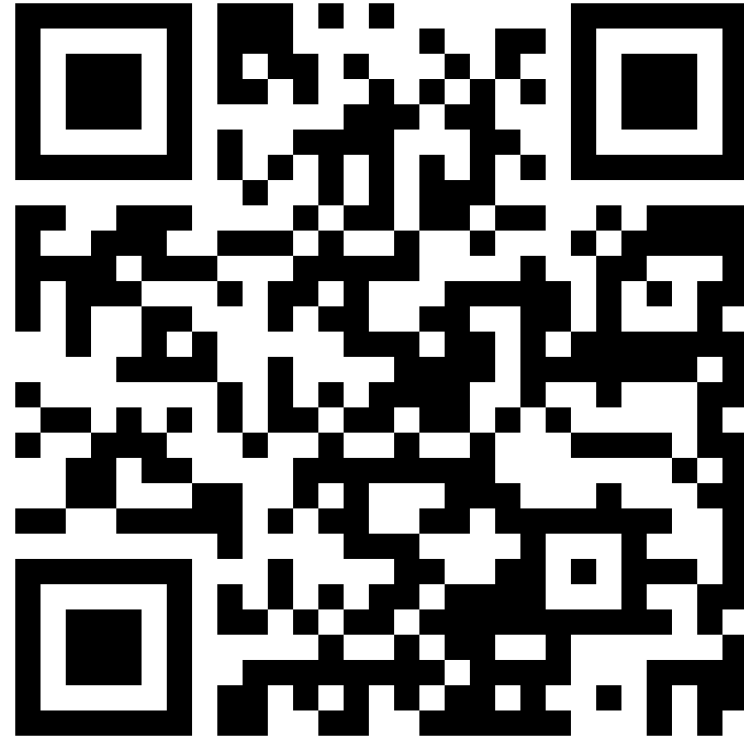
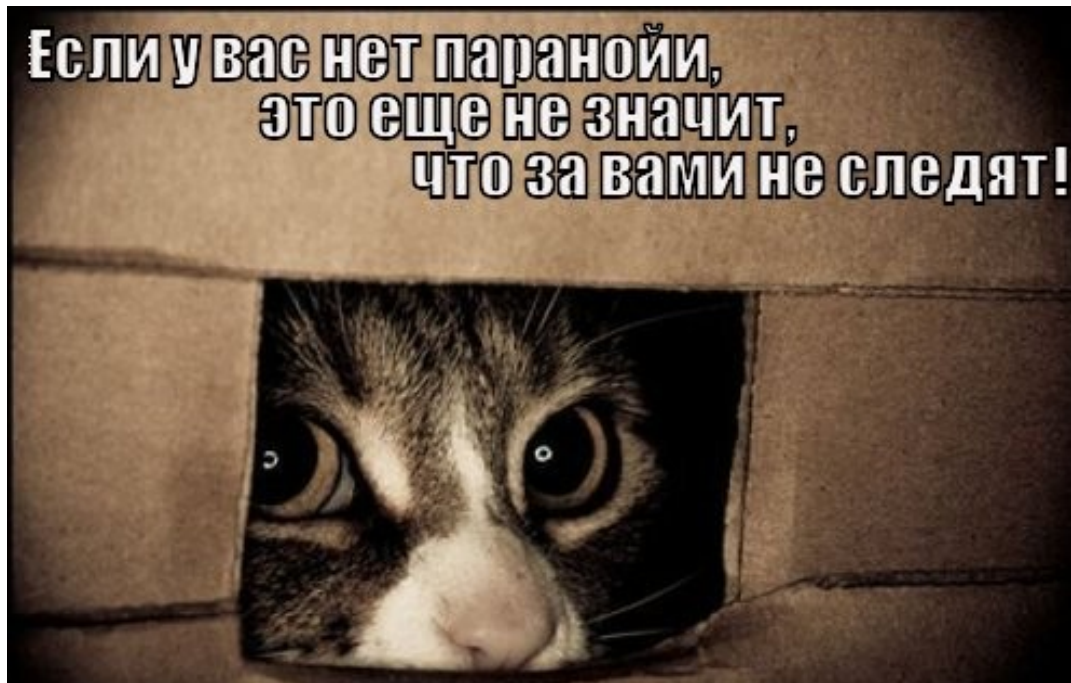


Зачем это все? Какой смысл?

РЕАЛИЗАЦИЯ «WHITE LIST» ДЛЯ ЗАГРУЗКИ ОС
И НИЗКОУРОВНЕВЫХ ПРИЛОЖЕНИЙ

ЗАЩИТА ОТ НИЗКОУРОВНЕВОГО ВРЕДОНОСНОГО ПО

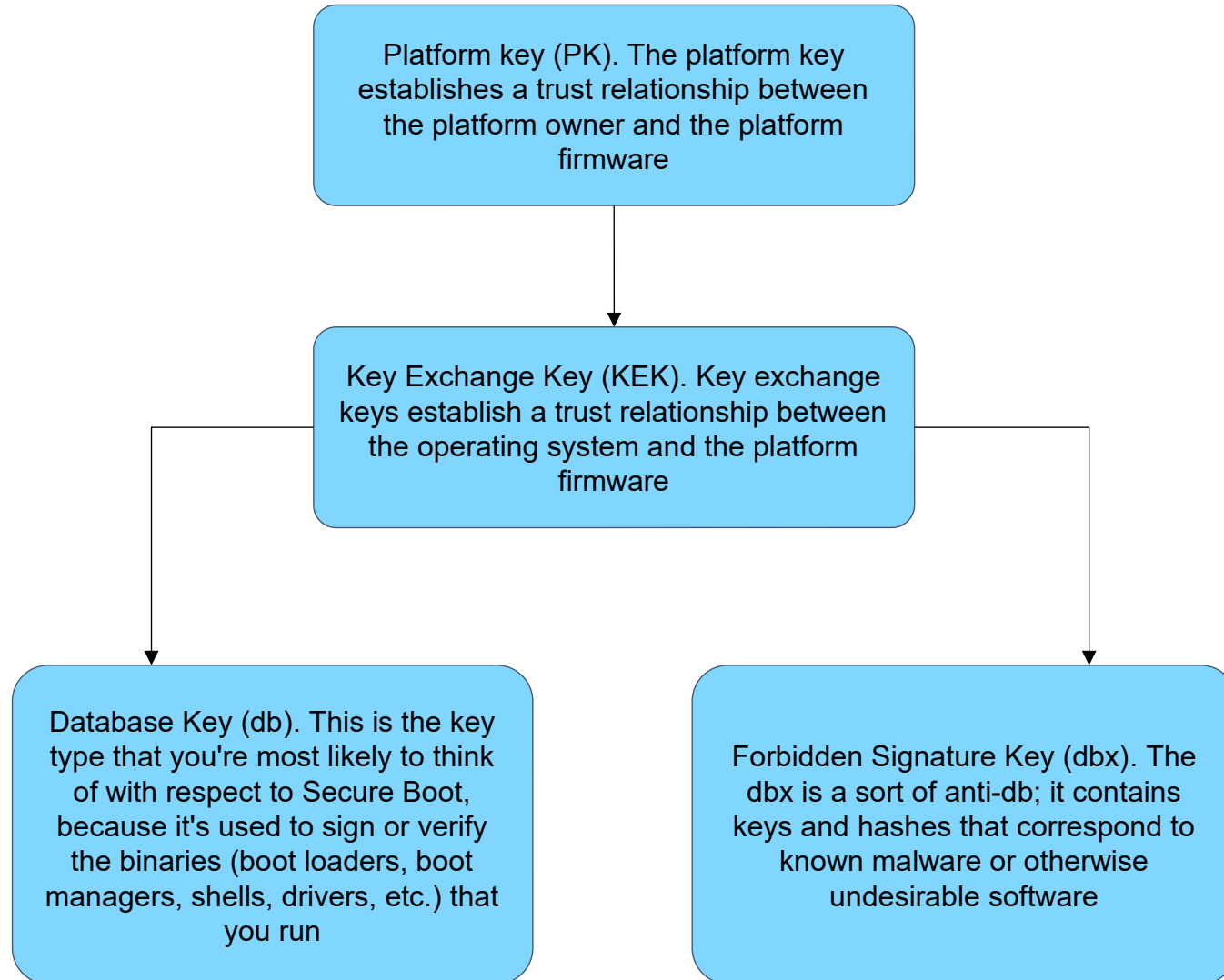
ЗАЩИТА ОТ «УТЕКШИХ» КЛЮЧЕЙ ВЕНДОРОВ



ЗДЕСЬ МОЖНО ПОЧИТАТЬ ПРО ЭКСПЛУАТАЦИЮ
ПОДПИСАННЫХ ЗАГРУЗЧИКОВ ДЛЯ ОБХОДА
SECURE BOOT

Настройка UEFI Secure Boot с own machine key.

Подготовка



ПОДРОБНЫЙ ГАЙД ПО SECURE BOOT
ОТ MICROSOFT

Настройка UEFI Secure Boot с own machine key.

Подготовка

ТЕКУЩИЕ КЛЮЧИ МОЖНО ПРОСМОТРЕТЬ
УТИЛИТОЙ EFI-READVAR

СОХРАНЯЕМ ТЕКУЩИЕ КЛЮЧИ:

```
for var in PK KEK db dbx; do efi-readvar -v $var -o old_${var}.esl; done
```

```
-rw----- 1 hideo hideo 5.0K Oct  9 01:56 old_db.esl
-rw----- 1 hideo hideo 16K Oct  9 01:56 old_dbx.esl
-rw----- 1 hideo hideo 2.5K Oct  9 01:56 old_KEK.esl
-rw----- 1 hideo hideo  0 Oct  9 01:56 old_PK.esl
```

```
hideo@archlinux ~$ efi-readvar
Variable PK, length 983
PK: List 0, type X509
  Signature 0, size 955, owner 3cc24e96-22c7-41d8-8863-8e39dc2cc2cf
  Subject:
    C=JP, ST=Kanagawa, L=Yokohama, O=Lenovo Ltd., CN=Lenovo Ltd. PK CA 2012
  Issuer:
    C=JP, ST=Kanagawa, L=Yokohama, O=Lenovo Ltd., CN=Lenovo Ltd. PK CA 2012
Variable KEK, length 2545
KEK: List 0, type X509
  Signature 0, size 957, owner 7facc7b6-127f-4e9c-9c5d-080f98994345
  Subject:
    C=JP, ST=Kanagawa, L=Yokohama, O=Lenovo Ltd., CN=Lenovo Ltd. KEK CA 2012
  Issuer:
    C=JP, ST=Kanagawa, L=Yokohama, O=Lenovo Ltd., CN=Lenovo Ltd. KEK CA 2012
KEK: List 1, type X509
  Signature 0, size 1532, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
  Subject:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Corporation KEK CA 2011
  Issuer:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Corporation Third Party Marketplace Root
Variable db, length 5080
db: List 0, type X509
  Signature 0, size 962, owner 7facc7b6-127f-4e9c-9c5d-080f98994345
  Subject:
    C=JP, ST=Kanagawa, L=Yokohama, O=Lenovo Ltd., CN=ThinkPad Product CA 2012
  Issuer:
    C=JP, ST=Kanagawa, L=Yokohama, O=Lenovo Ltd., CN=Lenovo Ltd. Root CA 2012
db: List 1, type X509
  Signature 0, size 919, owner 7facc7b6-127f-4e9c-9c5d-080f98994345
  Subject:
    C=US, ST=North Carolina, O=Lenovo, CN=Lenovo UEFI CA 2014
  Issuer:
    C=US, ST=North Carolina, O=Lenovo, CN=Lenovo UEFI CA 2014
db: List 2, type X509
  Signature 0, size 1572, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
  Subject:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Corporation UEFI CA 2011
  Issuer:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Corporation Third Party Marketplace Root
db: List 3, type X509
  Signature 0, size 1515, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
  Subject:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Windows Production PCA 2011
  Issuer:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Root Certificate Authority 2010
Variable dbx, length 16056
dbx: List 0, type X509
  Signature 0, size 1076, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
  Subject:
    C=GB, ST=Isle of Man, O=Canonical Ltd., OU=Secure Boot, CN=Canonical Ltd. Secure Boot Signing
  Issuer:
    C=GB, ST=Isle of Man, L=Douglas, O=Canonical Ltd., CN=Canonical Ltd. Master Certificate Authority
dbx: List 1, type X509
  Signature 0, size 784, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
```

Настройка UEFI Secure Boot с own machine key. Подготовка

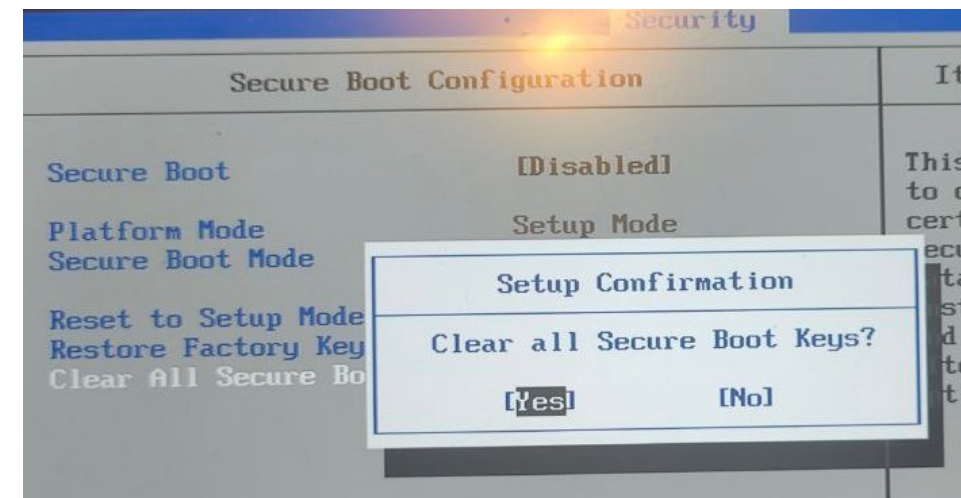
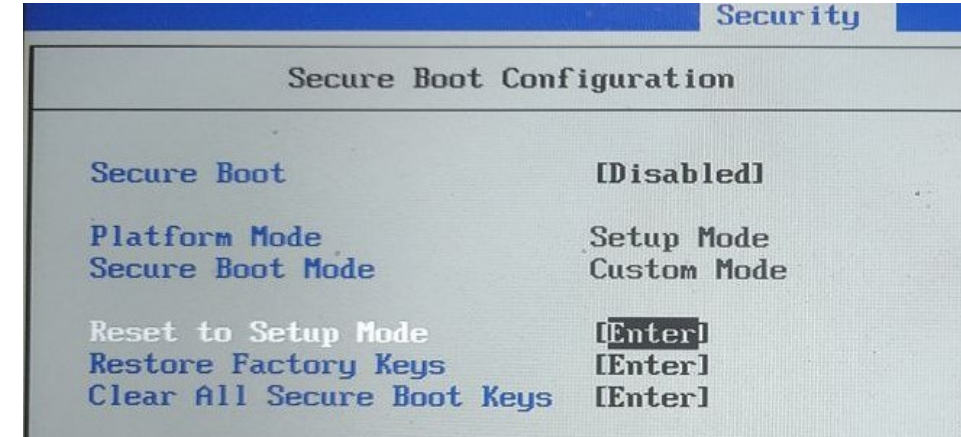
● ПЕРЕВОДИМ SECURE BOOT В РЕЖИМ НАСТРОЕК

ОЧИЩАЕМ ТЕКУЩИЕ КЛЮЧИ

МОНТИРУЕМ ЗАГРУЗОЧНЫЙ РАЗДЕЛ: MOUNT
/DEV/SDA2 /BOOT/EFI

ДЕЛАЕМ БЭКАП ЗАГРУЗОЧНОГО РАЗДЕЛА:
SUDO CP -R /BOOT/EFI /BOOT/EFI_BACKUP

УСТАНАВЛИВАЕМ НЕОБХОДИМЫЕ УТИЛИТЫ:
SUDO PIP3 INSTALL EFITOOLS SBSIGNTOOL



Настройка UEFI Secure Boot с own machine key. Создание ключей

```
# Создаем уникальный id  
uuidgen -r > guid.txt
```

```
# Конвертируем ключи, используя созданный UID  
cert-to-efi-sig-list -g "$(< guid.txt)" PK.crt PK.esl  
cert-to-efi-sig-list -g "$(< guid.txt)" KEK.crt KEK.esl  
cert-to-efi-sig-list -g "$(< guid.txt)" db.crt db.esl
```

```
# Подписываем списки сертификатов
```

```
# Подписываем PK самим собой
```

```
sign-efi-sig-list -g "$(< guid.txt)" -k PK.key -c PK.crt PK PK.esl PK.auth
```

```
# Подписываем KEK ключом PK
```

```
sign-efi-sig-list -g "$(< guid.txt)" -k PK.key -c PK.crt KEK KEK.esl  
KEK.auth
```

```
# Подписываем db ключом KEK
```

```
sign-efi-sig-list -g "$(< guid.txt)" -k KEK.key -c KEK.crt db db.esl  
db.auth
```

```
# Подписываем dbx ключом KEK
```

```
sign-efi-sig-list -g "$(< guid.txt)" -k KEK.key -c KEK.crt dbx  
old_dbx.esl dbx.auth
```

```
hideo@archlinux ~/uefi-keys$ sign-efi-sig-list -t "$(date --date='1 second' +%Y-%m-%d %H:%M:%S)" -k PK.key -c PK.crt PK PK.esl PK.auth  
Timestamp is 2023-10-12 22:47:05  
Authentication Payload size 1389  
Signature of size 1955  
Signature at: 40  
hideo@archlinux ~/uefi-keys$ sign-efi-sig-list -t "$(date --date='1 second' +%Y-%m-%d %H:%M:%S)" -k PK.key -c PK.crt KEK KEK.esl KEK.auth  
Timestamp is 2023-10-12 22:47:16  
Authentication Payload size 1399  
Signature of size 1955  
Signature at: 40  
hideo@archlinux ~/uefi-keys$ sign-efi-sig-list -t "$(date --date='1 second' +%Y-%m-%d %H:%M:%S)" -k KEK.key -c KEK.crt db ms_db.esl db.auth  
Timestamp is 2023-10-12 22:47:26  
Authentication Payload size 1453  
Signature of size 1967  
Signature at: 40
```

```
hideo@archlinux ~/uefi-keys$ lsa  
total 88K  
drwxr-xr-x  3 hideo hideo 4.0K Oct  9 02:02 .  
drwx----- 52 hideo hideo 4.0K Oct  9 02:02 ..  
-rw-----  1 hideo hideo 3.3K Oct  9 01:56 DB.auth  
-rw-r--r--  1 hideo hideo 1.3K Oct  9 01:56 DB.cer  
-rw-r--r--  1 hideo hideo 1.8K Oct  9 01:56 DB.crt  
-rw-r--r--  1 hideo hideo 1.4K Oct  9 01:56 DB.esl  
-rw-----  1 hideo hideo 3.2K Oct  9 01:56 DB.key  
-rw-----  1 hideo hideo 3.3K Oct  9 01:56 KEK.auth  
-rw-r--r--  1 hideo hideo 1.3K Oct  9 01:56 KEK.cer  
-rw-r--r--  1 hideo hideo 1.8K Oct  9 01:56 KEK.crt  
-rw-r--r--  1 hideo hideo 1.4K Oct  9 01:56 KEK.esl  
-rw-----  1 hideo hideo 3.2K Oct  9 01:56 KEK.key  
-rwxr-xr-x  1 hideo hideo 5.0K Oct  9 01:56 make_uefi_key.sh  
-rw-r--r--  1 hideo hideo   37 Oct  9 01:56 myGUID.txt  
-rw-----  1 hideo hideo 2.0K Oct  9 01:56 noPK.auth  
-rw-r--r--  1 hideo hideo   0 Oct  9 01:56 noPK.esl  
drwxr-xr-x  3 hideo hideo 4.0K Oct  9 02:02 old  
-rw-----  1 hideo hideo 3.3K Oct  9 01:56 PK.auth  
-rw-r--r--  1 hideo hideo 1.3K Oct  9 01:56 PK.cer  
-rw-r--r--  1 hideo hideo 1.8K Oct  9 01:56 PK.crt  
-rw-r--r--  1 hideo hideo 1.4K Oct  9 01:56 PK.esl  
-rw-----  1 hideo hideo 3.2K Oct  9 01:56 PK.key  
hideo@archlinux ~/uefi-keys$
```


Настройка UEFI Secure Boot с own machine key. Добавляем сертификаты Microsoft для загрузки Windows

```
# Скачиваем сертификат Copy
wget --user-agent="Mozilla"
https://www.microsoft.com/pkiops/certs/MicWinProPCA2011_2011-10-19.crt

wget --user-agent="Mozilla"
https://www.microsoft.com/pkiops/certs/MicCorUEFCA2011_2011-06-27.crt

# Добавляем GUID Microsoft в файл
echo "77fa9abd-0359-4d32-bd60-28f4e78f784b" > msguid.txt

# Конвертируем в формат .esl
sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output ms_win_db.esl MicWinProPCA2011_2011-10-19.crt

sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output ms_uefi_win_db.esl MicCorUEFCA2011_2011-06-27.crt

# Подписываем списки сертификатов
sign-efi-sig-list -a -g 77fa9abd-0359-4d32-bd60-28f4e78f784b -k KEK.key -c KEK.crt db ms_win_db.esl add_ms_db.auth

sign-efi-sig-list -a -g 77fa9abd-0359-4d32-bd60-28f4e78f784b -k KEK.key -c KEK.crt db ms_uefi_win_db.esl add_ms_uefi_db.auth
```

```
root@archlinux /home/hideo/uefi-keys$ wget --user-agent="Mozilla" https://www.microsoft.com/pkiops/cer
--2023-10-13 02:58:19-- https://www.microsoft.com/pkiops/certs/MicWinProPCA2011_2011-10-19.crt
Loaded CA certificate '/etc/ssl/certs/ca-certificates.crt'
Resolving www.microsoft.com (www.microsoft.com)... 95.101.145.130, 2a02:26f0:9500:c92::356e, 2a02:26f0
Connecting to www.microsoft.com (www.microsoft.com)|95.101.145.130|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1499 (1.5K) [application/octet-stream]
Saving to: 'MicWinProPCA2011_2011-10-19.crt'

MicWinProPCA2011_2011-10-19.crt          100%[=====]
2023-10-13 02:58:20 (17.0 MB/s) - 'MicWinProPCA2011_2011-10-19.crt' saved [1499/1499]

root@archlinux /home/hideo/uefi-keys$
root@archlinux /home/hideo/uefi-keys$
root@archlinux /home/hideo/uefi-keys$ sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x5
root@archlinux /home/hideo/uefi-keys$ sign-efi-sig-list -a -g 77fa9abd-0359-4d32-bd60-28f4e78f784b -k
Timestamp is 0-0-0 00:00:00
Authentication Payload size 1583
Signature of size 1967
Signature at: 40
```

Настройка UEFI Secure Boot с own machine key. Добавление ключей в систему

```
# Вводим свой пароль для BIOS
sudo cat > /sys/class/firmware-attributes \
/thinklmi/authentication/Admin/current_password
my-super-secret-password
^D

# Создаем директории для ключей
mkdir -p /etc/secureboot/keys/{db,dbx,KEK,PK}
cp PK.auth /etc/secureboot/keys/PK
cp KEK.auth /etc/secureboot/keys/KEK
cp db.auth /etc/secureboot/keys/db
cp add_ms_db.auth /etc/secureboot/keys/db
cp dbx.auth /etc/secureboot/keys/dbx
|

# Загружаем ключи в систему
sbkeysync --verbose

# Загружаем Platform key
efi-updatevar -f /etc/secureboot/keys/PK/PK.auth PK
```

```
root@archlinux /home/hideo/uefi-keys$ tree /etc/secureboot/keys
/etc/secureboot/keys
├── db
│   └── db.auth
├── dbx
│   └── dbx.auth
├── KEK
│   └── KEK.auth
└── PK
    └── PK.auth

5 directories, 4 files
```


Настройка UEFI Secure Boot с own machine key. Проверка загруженных ключей

ЧИТАЕМ ПЕРЕМЕННЫЕ УТИЛИТОЙ EFI-READVAR
ИЗ РАНЕЕ УСТАНОВЛЕННОГО КОМПЛЕКТА

УБЕЖДАЕМСЯ, ЧТО ВСЕ КЛЮЧИ УСПЕШНО
ЗАГРУЗИЛИСЬ В СИСТЕМУ

```
root@archlinux /home/hideo/uefi-keys$ efi-readvar
Variable PK, length 1349
PK: List 0, type X509
  Signature 0, size 1321, owner babc8322-ae90-40fd-a776-aae07d70eb38
  Subject:
    CN=My Platform Key
  Issuer:
    CN=My Platform Key
Variable KEK, length 1357
KEK: List 0, type X509
  Signature 0, size 1329, owner babc8322-ae90-40fd-a776-aae07d70eb38
  Subject:
    CN=My Key Exchange Key
  Issuer:
    CN=My Key Exchange Key
Variable db, length 2912
db: List 0, type X509
  Signature 0, size 1341, owner babc8322-ae90-40fd-a776-aae07d70eb38
  Subject:
    CN=My Signature Database key
  Issuer:
    CN=My Signature Database key
db: List 1, type X509
  Signature 0, size 1515, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
  Subject:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Windows Production PCA 2011
  Issuer:
    C=US, ST=Washington, L=Redmond, O=Microsoft Corporation, CN=Microsoft Root Certificate Authority 2010
Variable dbx, length 16056
dbx: List 0, type X509
  Signature 0, size 1076, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
  Subject:
    C=GB, ST=Isle of Man, O=Canonical Ltd., OU=Secure Boot, CN=Canonical Ltd. Secure Boot Signing
  Issuer:
    C=GB, ST=Isle of Man, L=Douglas, O=Canonical Ltd., CN=Canonical Ltd. Master Certificate Authority
dbx: List 1, type X509
  Signature 0, size 784, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
  Subject:
    CN=Debian Secure Boot Signer
  Issuer:
    CN=Debian Secure Boot CA
dbx: List 2, type SHA256
  Signature 0, size 48, owner 77fa9abd-0359-4d32-bd60-28f4e78f784b
```

Настройка UEFI Secure Boot с own machine key. Подписка загрузчиков

```
hideo@archlinux ~/uefi-keys$  
hideo@archlinux ~/uefi-keys$ sudo sbsign --key db.key --cert db.crt --output /boot/efi/EFI/arch/arch.efi /boot/efi/EFI/arch/arch.efi  
Signing Unsigned original image  
hideo@archlinux ~/uefi-keys$ sudo sbsign --key db.key --cert db.crt --output /boot/efi/EFI/refind/refind.efi /boot/efi/EFI/refind/refind.efi  
warning: data remaining[301568 vs 331150]: gaps between PE/COFF sections?  
warning: data remaining[301568 vs 331152]: gaps between PE/COFF sections?  
Signing Unsigned original image  
hideo@archlinux ~/uefi-keys$ sudo sbsign --key db.key --cert db.crt --output /boot/efi/EFI/Boot/bootx64.efi /boot/efi/EFI/Boot/bootx64.efi  
Image was already signed; adding additional signature  
hideo@archlinux ~/uefi-keys$
```

```
sudo mount /dev/sda2 /boot/efi
```

```
sudo sbsign --key db.key --cert db.crt \  
--output /boot/efi/EFI/refind/refind.efi \  
/boot/efi/EFI/refind/refind.efi
```

```
sudo sbsign --key db.key --cert db.crt \  
--output /boot/efi/EFI/arch/arch.efi \  
/boot/efi/EFI/arch/arch.efi
```

Настройка UEFI Secure Boot с own machine key. Проверка настроек BIOS

Secure Boot Configuration

Secure Boot

[Enabled]

Platform Mode

User Mode

Secure Boot Mode

Custom Mode

Reset to Setup Mode

[Enter]

Restore Factory Keys

[Enter]

Clear All Secure Boot Keys

[Enter]

Настройка UEFI Secure Boot с own machine key. Проверка невозможности загрузки неподписанных .efi приложений

rEFInd - MOK utility at EFI\tools\HashTool.efi on EFI system partition

Starting HashTool.efi

Using load options "

Secure Boot validation failure loading HashTool.efi!

This computer is configured with Secure Boot active, but HashTool.efi has failed validation.

You can:

- * Launch another boot loader
- * Disable Secure Boot in your firmware
- * Sign HashTool.efi with a machine owner key (MOK)
- * Use a MOK utility (often present on the second row) to add a MOK with which HashTool.efi has already been signed.
- * Use a MOK utility to register HashTool.efi ("enroll its hash") without signing it.

See <http://www.rodsbooks.com/refind/secureboot.html> for more information

* Hit any key to continue *

[01]

Используем Secure Boot в Linux на всю катушку
<https://habr.com/ru/articles/308032/>

[02]

О безопасности UEFI, часть пятая
<https://habr.com/ru/articles/267953/>

[03]

«Укрощаем» UEFI Secure Boot
<https://habr.com/ru/articles/273497/>

[04]

Полнодисковое шифрование Windows
и Linux установленных систем.
Зашифрованная мультизагрузка
<https://habr.com/ru/articles/482696/>

[05]

Hardware-and-Firmware-Security-
Guidance
<https://github.com/nsacyber/Hardware-and-Firmware-Security-Guidance>

[06]

Документация VeraCrypt
<https://www.veracrypt.fr/en/Documentation.html>





Больше
практических кейсов,
результатов исследований
от [Solar 4RAYS](#)

INPUT = {PARANOID_AND.COMPLICATED}