Šifrování disků a TrueCrypt

Milan Brož
xbroz@fi.muni.cz

EurOpen 2013
Vranov nad Dyjí / PA168

TrueCrypt

- **transparentní šifrování disku**
  FDE - Full Disk Encryption
- **multiplatformní**
  Windows, Linux, MacOS
- původně odvozeno od E4M
  Encryption for Masses
- Linux - používá nativní služby jádra
  dm-crypt vs FUSE ("FS in Userspace")
- Windows - podpora šifrovaného OS
  vlastní bootloader

TrueCrypt licence

- Open Source
- **nestandardní licence**
  TrueCrypt License Version x.y
- **nekompatibilní s Linux distribucemi**
  požadavek na čisté licence
- nejasný pokus o GPL ve verzi 2.0 ...
  "Released under the original E4M license to avoid potential problems relating to the GPL license." (2.1 Readme.txt)
- Licenční text změněn 21×
  verze 1.0 - 7.1a, bez nepodstatných změn

Alternativní implementace (Linux)

- **RealCrypt**
  nezávislý překlad z původních zdrojových kódů
- **ScramDisk**
  vyžaduje vlastní jaderný ovladač
- **tc-play**
  - BSD license, původně pro DragonFly OS
  - používá jaderný dm-crypt

...

- **cryptsetup**

Metadata – hlavička na disku

- **vždy šifrovaná**
  nelze rozpoznat bez znalosti klíče

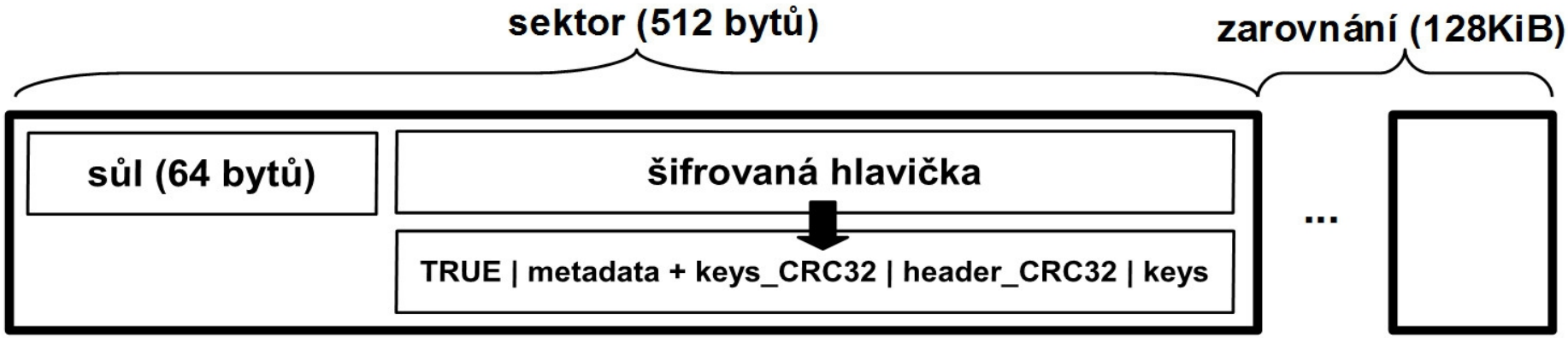
- **klíč odvozen z hesla pomocí PBKDF2**
  - 1000/2000 iterací
  - hash RMD160, SHA1, SHA512, Whirlpool

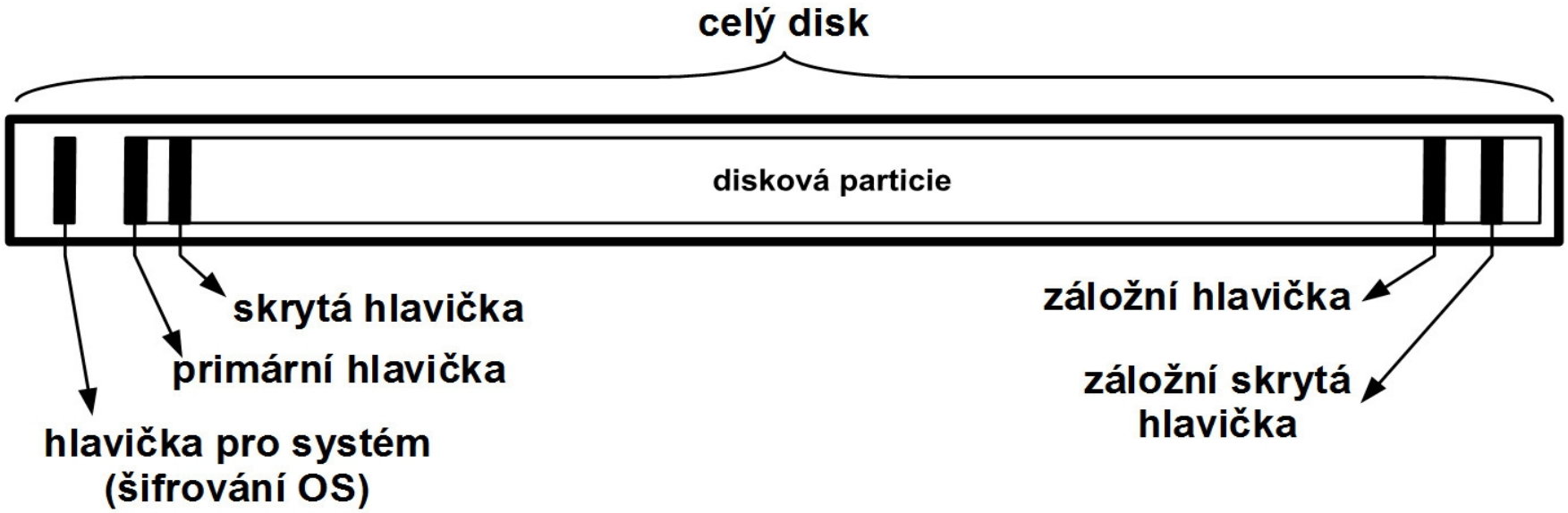
- **iterace přes všechny známé módy**
  hledá řetězec "TRUE" a správný CRC32
  stejný mód je pak použit pro data

Metadata – hlavička na disku

```
# cryptsetup tcryptDump tc.img --debug
...
Enter passphrase:
...
# Trying to load TCRYPT crypt type from device tc.img.
# Crypto backend (gcrypt 1.5.2) initialized.
# Reading TCRYPT header of size 512 bytes from device tc.img.
# TCRYPT: trying KDF: pbkdf2-ripemd160-2000.
# TCRYPT:  trying cipher aes-xts-plain64
# TCRYPT:  trying cipher serpent-xts-plain64
# TCRYPT:  trying cipher twofish-xts-plain64
# TCRYPT:  trying cipher twofish-aes-xts-plain64
...
# TCRYPT: trying KDF: pbkdf2-sha512-1000.
# TCRYPT:  trying cipher aes-xts-plain64
# TCRYPT:  trying cipher serpent-xts-plain64
# TCRYPT:  trying cipher twofish-xts-plain64
# TCRYPT:  trying cipher twofish-aes-xts-plain64
# TCRYPT:  trying cipher serpent-twofish-aes-xts-plain64
# TCRYPT: Signature magic detected.
# TCRYPT: Header version: 5, req. 7, sector 512, mk_offset 131072,
hidden_size 0, volume size 33292288
# TCRYPT: Header cipher serpent-twofish-aes-xts-plain64, key size 192
...
```

Metadata – hlavička na disku





Šifrovací algoritmy a módy

Algoritmus	Blok [bitů]	Klíč [bitů]	Mód	Zavedeno [verze]	Zrušeno [verze]
AES256	128	256	XTS	5.0	-
"	"	"	LRW	4.1	5.0
"	"	"	CBC	2.0	4.1
Serpent	128	256	XTS	5.0	-
"	"	"	LRW	4.1	5.0
"	"	"	CBC	3.0	4.1
Twofish	128	256	XTS	5.0	-
"	"	"	LRW	4.1	5.0
"	"	"	CBC	3.0	4.1
Blowfish (LE)	64	448	LRW	4.1	4.3
"	"	"	CBC	1.0	4.1
CAST5	64	128	LRW	4.1	4.3
"	"	"	CBC	1.0	4.1
TripleDES (EDE)	64	168	LRW	4.1	4.3
"	"	"	CBC	1.0	4.1
IDEA	64	128	CBC	1.0	2.1a

Zřetězené šifry

- kaskáda algoritmů
- ochrana před prolomením jednoho algoritmu
  algoritmy se nesmí vzájemně ovlivňovat
- podstatné snížení propustnosti

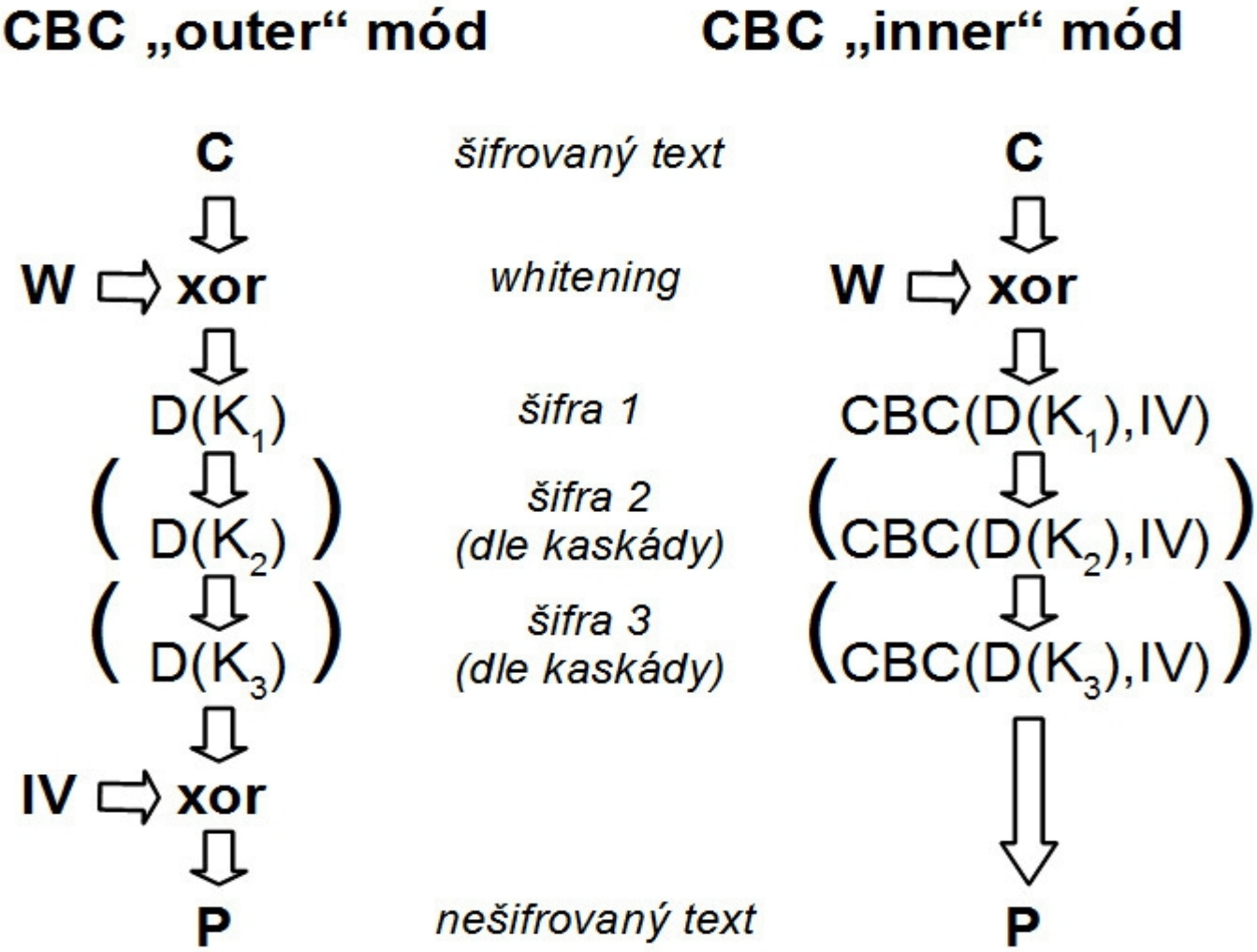
```
# lsblk /dev/loop0 -o NAME,TYPE
loop0      loop
└─tc_2     crypt  -> aes-xts-plain64
  └─tc_1   crypt  -> twofish-xts-plain64
    └─tc   crypt  -> serpent-xts-plain64

# cryptsetup status tc
/dev/mapper/tc is active.
  type:    TCRYPT
  cipher:  serpent-twofish-aes-xts-plain64
  keysize: 1536 bits
  device:  /dev/loop0
  loop:    /test/tc.img
  offset:  256 sectors
  size:    65024 sectors
  skipped: 256 sectors
  mode:    read/write
```

```
# truecrypt --volume-properties /mnt/tst
Slot: 1
Volume: /test/tc.img
Virtual Device: /dev/mapper/truecrypt1
Mount Directory: /mnt/tst
Size: 31.8 MB
Type: Normal
Read-Only: No
Hidden Volume Protected: No
Encryption Algorithm: AES-Twofish-Serpent
Primary Key Size: 768 bits
Secondary Key Size (XTS Mode): 768 bits
Block Size: 128 bits
Mode of Operation: XTS
PKCS-5 PRF: HMAC-SHA-512
Volume Format Version: 2
Embedded Backup Header: Yes
```

Zřetězené šifry v CBC módu

- nutnost řešit problém různé velikosti bloku



Generátor náhodných čísel (RNG)

- klíče, IV semínko, whitening, sůl
- **interní implementace RNG**
- **zdroje entropie** (mixovány do interního poolu)
  - pohyb kurzoru
  - klávesnice
  - Windows: interní statistiky systému
  - Linux/MacOS: /dev/[u]random

Skrytý disk a skrytý OS

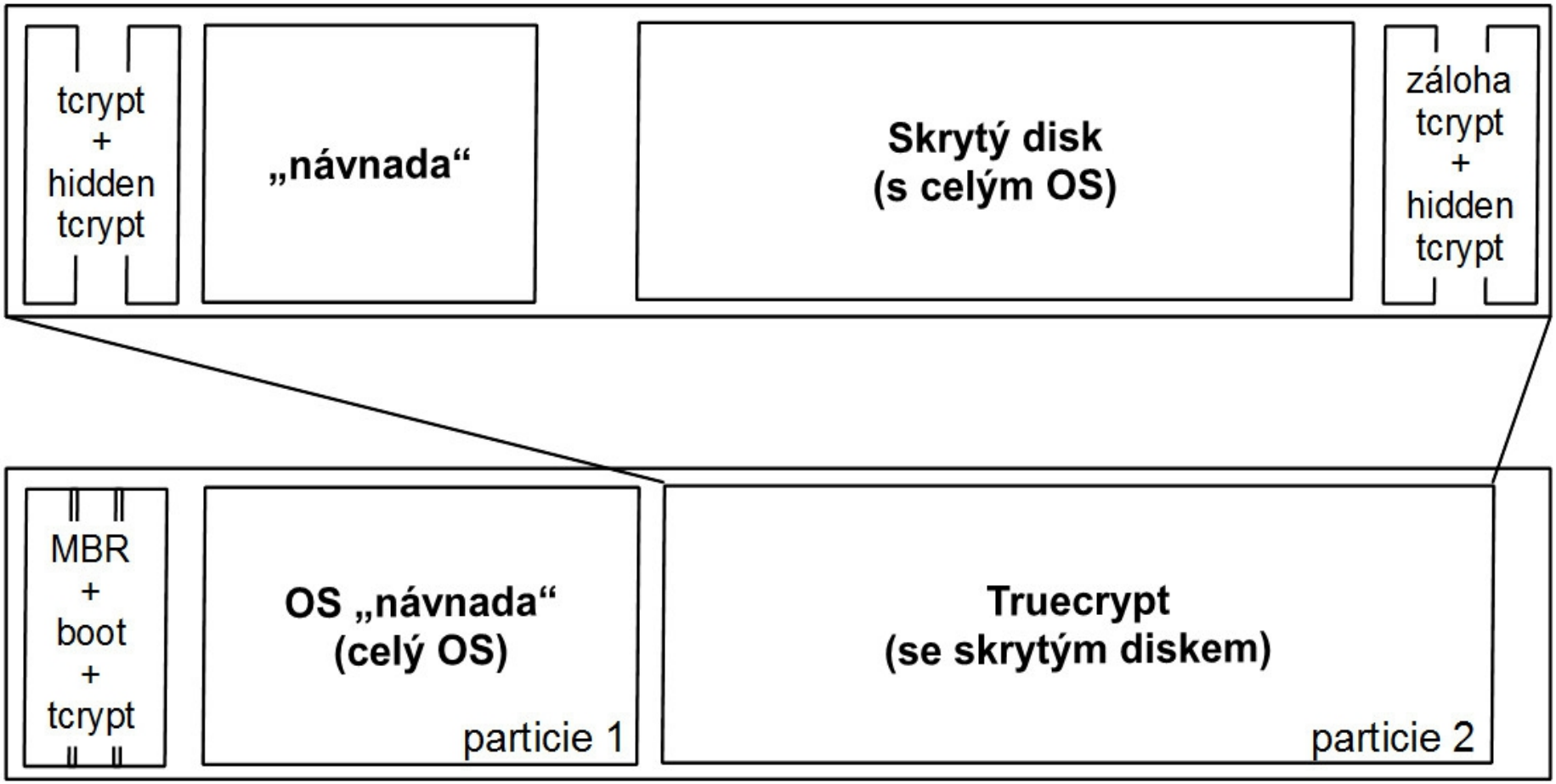
- důvěryhodné popření (existence dat)
  "Plausible deniability"

- **alokační strategie FAT souborového systému**
  - skrytý disk v nevyužité části
  - problematická ochrana před přepsáním

- fixní umístění hlavičky skrytého disku

- **prosakování informací / existence disku**
  => *skrytý operační systém*

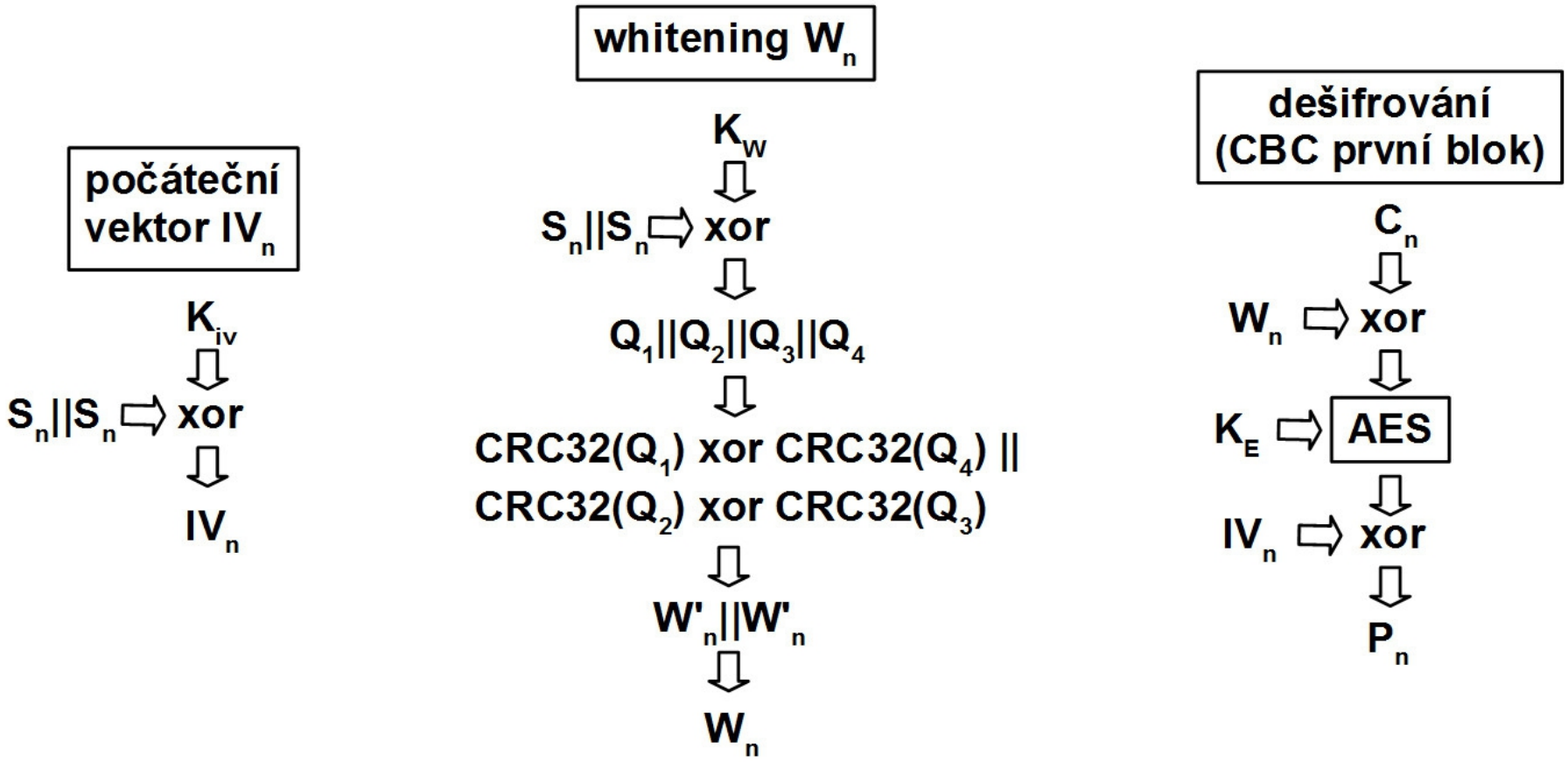
Skrytý disk a skrytý OS



CBC mód – útok na skrytý disk

Cílem je prokázat existenci skrytého disku

- klíče K_E, K_{IV}, K_W jsou uloženy v hlavičce
- první blok sektoru číslo S_n se dešifruje takto:



CBC mód – útok na skrytý disk

- pro $\mathbf{S_n}$ dělitelné 4 platí $\mathbf{S_n \text{ xor } S_{n+1} = S_{n+2} \text{ xor } S_{n+3}}$
- pro **W** platí $\mathbf{W(K_w, A \text{ xor } B) = W(K_w, A) \text{ xor } W(K_w, B)}$
- pozměňme první bloky sektorů n .. n+3

  $\mathbf{P_n}$ a $\mathbf{P_{n+2}} = \mathbf{0}$

  $\mathbf{P_{n+1}}$ a $\mathbf{P_{n+3}} = \mathbf{S_n \text{ xor } S_{n+1}}$

$C_n = E(K_E, P_n \text{ xor } IV_n) \text{ xor } W_n = \mathbf{E(K, K_{iv} \text{ xor } S_n)} \text{ xor } W_n$

$C_{n+1} = E(K_E, P_{n+1} \text{ xor } IV_{n+1}) \text{ xor } W_{n+1} = .. = \mathbf{E(K, K_{iv} \text{ xor } S_n)} \text{ xor } W_{n+1}$

- což umoňuje eliminovat vliv šifrování (xor) a platí

$\mathbf{C_n \text{ xor } C_{n+1} = C_{n+2} \text{ xor } C_{n+3}}$

Detekce značky (watermark) pouze z šifrovaného textu!

Co je špatně?

- **IV** je odvozen z tajného klíče, ale ...
  je predikovatelný *(např. v rámci sektorů po sobě)*
- **Whitening používá CRC32 místo hash funkce**
  nevhodné vlastnosti pro tyto účely
- ve verzi 4.1 **přechod na mód LRW**
  odstranění obou problémů výše
  ... za cenu použití příliš "nového" módu
- ... a LRW se nestal standardem
  design je dle IEEE problematický
  problém s odhalením LRW tweak key z hlavičky
- ... následný **přechod na mód XTS** (dle IEEE 1619)

Keyfiles

- přimíchání obsahu souborů k heslu
- použito jen 1 MiB obsahu souboru
- nevhodně opět postaveno na CRC32
  vliv keyfile lze eliminovat obsahem keyfile!
  (... útok předpokládá porušení security modelu)
  i v současné verzi
- vhodné pro tokeny
  tzn. k odemčení je potřeba token + heslo
- interní pool limituje heslo na max. 64 znaků

Cryptsetup & TrueCrypt formát

- separace crypto primitiv do knihoven / kernelu
- jednotná knihovna
  plain, LUKS, loop-AES, Truecrypt
- integrace v systemd
- (záměrně) neumožňuje modifikace Truecrypt hlavičky
- limitováno kernel crypto API
  - není podpora pro whitening + IV a některých šifer
  - plná podpora pro LRW a XTS mód
- podpora skrytého disku, kaskád

Cryptsetup & TrueCrypt formát

- man cryptsetup :-)
- informace z hlavičky **tcryptDump**

```
# cryptsetup tcryptDump tst

Enter passphrase:
TCRYPT header information for tst
Version:        5
Driver req.:    7
Sector size:    512
MK offset:      131072
PBKDF2 hash:    sha512
Cipher chain:   serpent-twofish-aes
Cipher mode:    xts-plain64
MK bits:        1536
```

Cryptsetup & TrueCrypt formát

- aktivace zařízení **tcryptOpen**

```
# cryptsetup tcryptOpen tst tcrypt_dev
Enter passphrase:
```

... vytvořeno zařízení /dev/mapper/tcrypt_dev

- stav zařízení **status**

```
# cryptsetup status tcrypt_dev
/dev/mapper/tcrypt_dev is active.
type:    TCRYPT
cipher:  serpent-twofish-aes-xts-plain64
keysize: 1536 bits
device:  /dev/loop0
loop:    /tmp/tst
offset:  256 sectors
size:    65024 sectors
skipped: 256 sectors
mode:    read/write
```

děkuji za pozornost