SECURITY VULNERABILITIES OF DIGITAL VIDEO BROADCAST CHIPSETS

Adam Gowdiak
Security Explorations

INTRODUCTION

About Security Explorations

- Security start-up company from Poland
- Provides various services in the area of security and vulnerability research
- Commercial and Pro Bono research projects
- Came to life in a result of a true passion of its founder for breaking security of things and analyzing software for security defects
- Our ambition is to conduct quality, unbiased, vendor-free and independent security and vulnerability research

INTRODUCTION

Presentation Goal

- Continuation of our research in a digital satellite TV area
- Educate about security risks associated with less known technologies and platforms such as those used in a digital satellite TV ecosystem
- Warn about security risks associated with
  - closed ecosystems such as digital satellite TV
  - insecurely implemented proprietary hardware components
  - 3rd party security evaluation processes

INTRODUCTION

DISCLAIMER

- Information provided in this presentation is for **educational purposes only**
- Security Explorations neither promotes, nor encourages the acts of a digital satellite TV piracy
- Any use of the information provided in this presentation for illegal purposes is strictly prohibited
- In case of legal actions taken against Security Explorations, the following web pages will be updated

  http://www.security-explorations.com/en/legal-threats.html

DIGITAL SATELLITE TV

Why bother about content security ?

- Pay TV piracy remains a major concern for channels and operators
  - it leads to financial losses for the European pay TV industry
  - it substantially damages the image of transmitters and content rights holders
  - it reduces the allure and payback of investing in the industry
  - it hurts the industry and its innovation capabilities
- Signal theft estimated to be more than $2.1 billion at the end of 2011for Asia region alone (CASBAA)

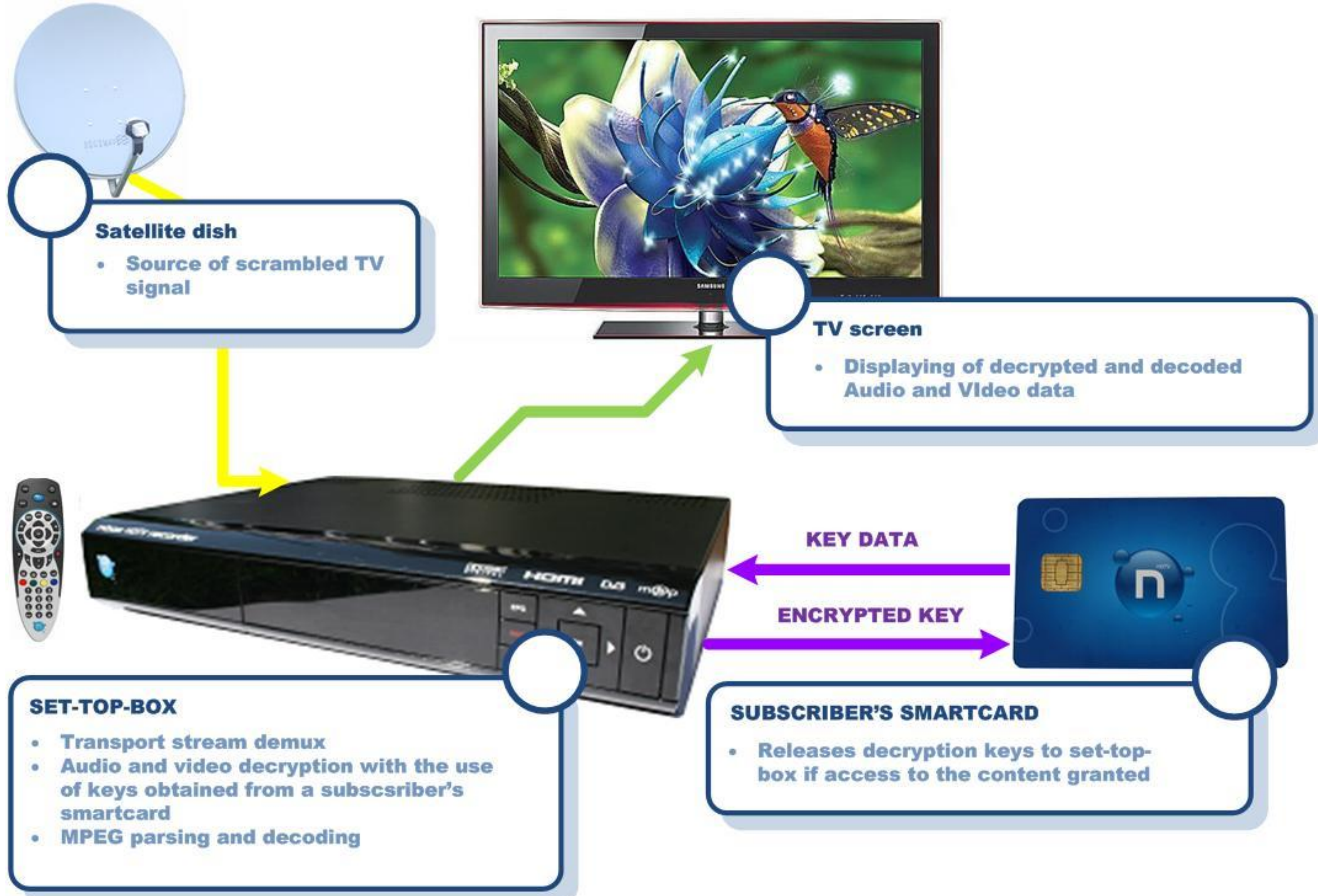
DIGITAL SATELLITE TV

Security of a premium content

- Paid, premium content broadcasted in encrypted form
  - Scrambling at the TS or PES level
    - `transport_scrambling_control` bit of MPEG TS packet
  - Common Scrambling Algorithm (CSA) and its derivatives
    - Shared 64-bit secret key (Control Word)
  - Dedicated security chipsets for decryption
- Key components in the security system
  - Subscriber's smartcard
    - holds information about subscriber's access rights to programming
    - releases decryption keys to the set-top-box if access to a given service is granted
  - Set-top-box
    - Conducts decryption of a scrambled content with the use of a received decryption key

DIGITAL SATELLITE TV

Security of a premium content (2)



DIGITAL SATELLITE TV

Control Words (CW)

- 64-bit secret keys used to descramble encrypted MPEG streams
  - Audio, video and data
- Unique to each programming
- Generated automatically by the content provider
  - Changed every ~10s
  - Odd and even keys for uninterrupted programming reception
    - Current and next key
- Broadcasted in encrypted form to client devices (set-top-boxes)
  - carried in entitlement control messages (ECM)
  - encrypted with the use of asymmetric crypto (i.e. RSA)

DIGITAL SATELLITE TV

Entitlement Control Messages (ECM)

- ECM messages contain private conditional access information such as Control Words
  - Broadcasted by the means of a dedicated MPEG stream
  - Message format specific to CAS vendor
- PID of MPEG stream carrying ECM messages denoted by `CA_descriptor`
  - If elementary stream is scrambled, a CA descriptor shall be present for the program containing that elementary stream
  - Usually present in `TS_program_map_section`
    - MPEG `table_id` = 0x02

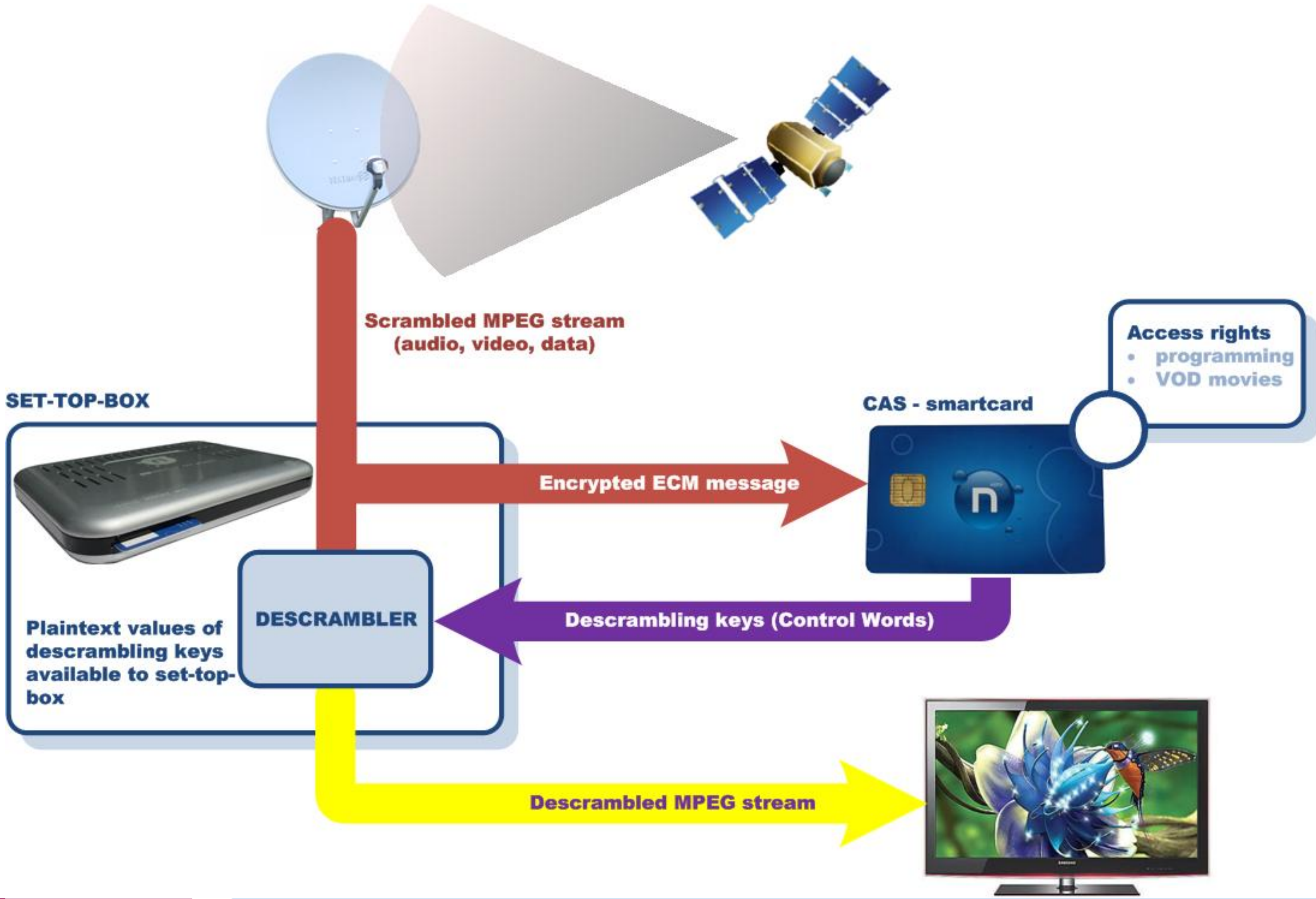
DIGITAL SATELLITE TV

Conditional Access System (CAS)

- It protects the content by requiring certain criteria to be met before granting access to the content
- Subscriber's smartcard holds information about subscriber's access rights to a given programming
  - what programming / program packages a subscriber is entitled to watch
- Only authorized client devices (paying subscribers) can decrypt MPEG streams for premium content
  - Set-top-box device asks the smartcard to decrypt encrypted Control Word (ECM message)
  - The smartcard makes sure that access to the content can be granted and releases the plaintext value of a Control Word

DIGITAL SATELLITE TV

CAS architecture (set-top-box side)



DIGITAL SATELLITE TV

Threats to the model

- Premium content is encrypted and broadcasted to all subscribers with the use of same crypto key (Control Word)
- One rogue subscriber with access to all premium content can share Control Word keys with others over the Internet
  - illegal reception / distribution of premium programming aka signal theft
  - Control Words sharing

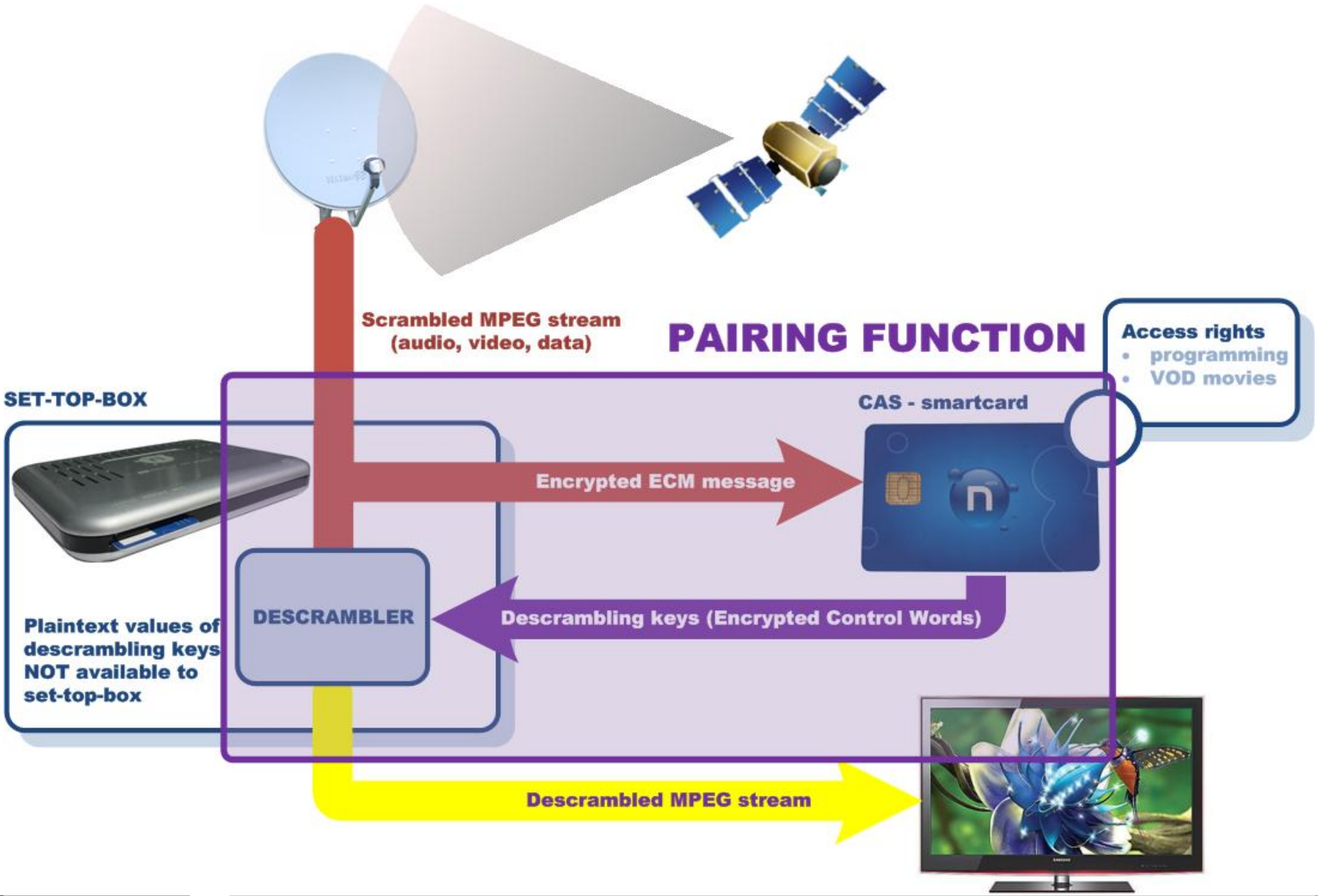
DIGITAL SATELLITE TV

CAS with chipset pairing

- Control Words unique for each pair of a subscriber (smartcard) / client device (set-top-box)
  - smart cards can be used only with secure devices
  - the link between the smart card and the client device is secured
  - illegal content redistribution is prevented (no more CW sharing)
- Chipset pairing has a form of a cryptographic function
- It is usually implemented in a silicon chip (DVB chipset)

DIGITAL SATELLITE TV

CAS with chipset pairing (set-top-box side)



DIGITAL SATELLITE TV

Pairing function

- A function that cryptographically ties a set-top-box device and a subscriber's smartcard

$$encCW = PAIRING_{enc}(CW, CWPK)$$

$$CW = PAIRING_{dec}(encCW, CWPK)$$

- Control Words pairing key (CWPK)
  - Unique to each subscriber
  - Assigned to it at the time of activating a given user's digital satellite TV subscription
  - Usually, a function of a unique DVB chipset's key

DIGITAL SATELLITE TV

Conax CAS with chipset pairing

- Conax AS is one of the major CAS providers for the Pay TV industry
  - More than 350 installations in 80 countries world-wide
- CAS implemented in software and hardware
  - Partnership with set-top-box vendors
  - Partnership with many DVB chipset vendors to implement chipset pairing functionality
    - STMicroelectronics, Broadcom, Renesas Electronics, …

DIGITAL SATELLITE TV

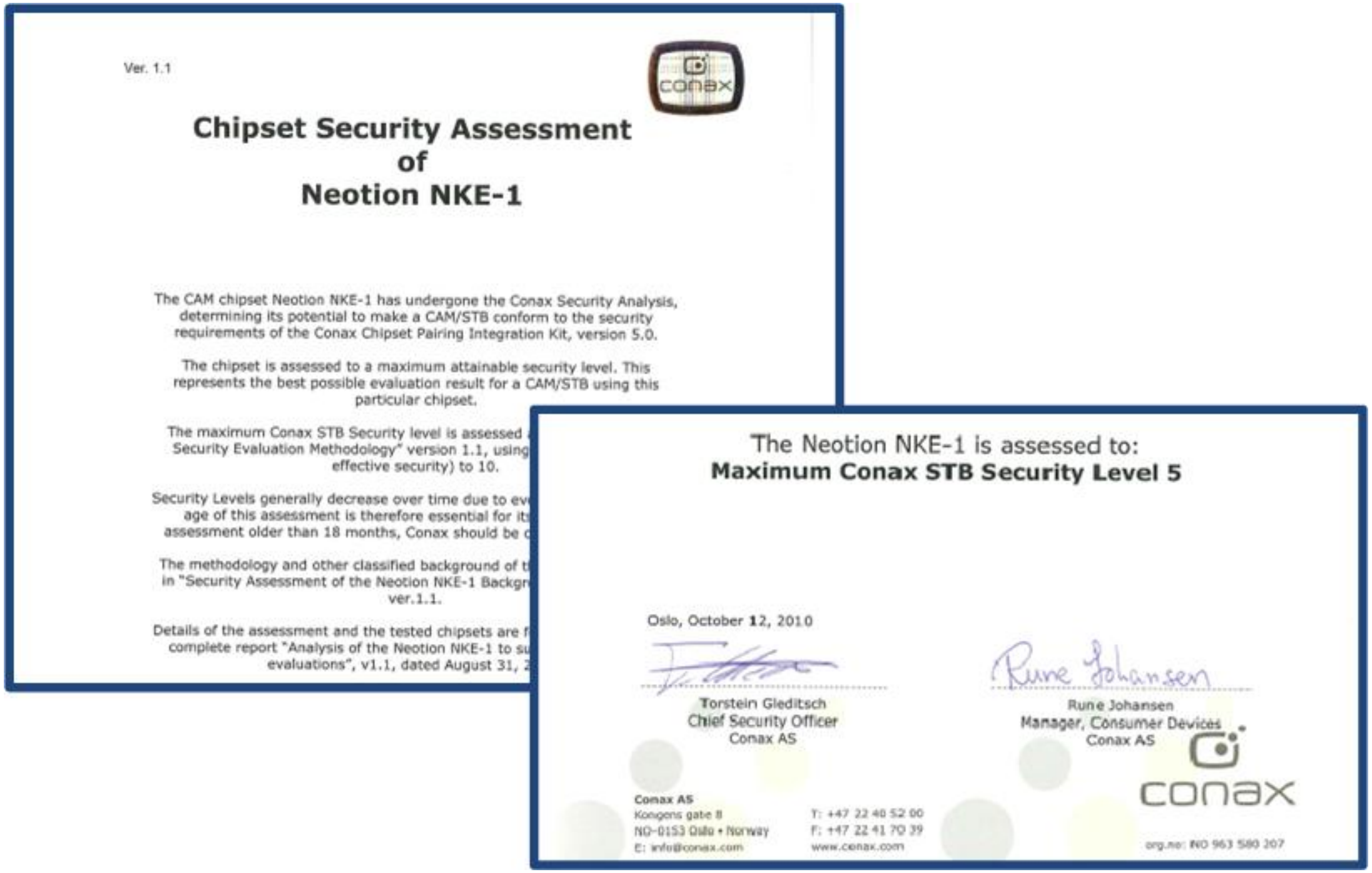
Conax security evaluation

- Mandatory, comprehensive evaluation of all qualified chipsets run at independent, world-leading security laboratories
  - All set-top-boxes and DVB chipsets implementing Conax CAS with chipset pairing undergo rigorous security evaluation process
  - Official scoring assigned to set-top-boxes and DVB chipsets and certified in writing by Conax CSO
    - „0 represents no security and 9 corresponds to the security level of Conax smart cards"

Source: http://www.conax.com/en/solutions/clientdevicesecurity/
Conax Security Department (09-Jan-2012)

DIGITAL SATELLITE TV

Conax security certification



Ver. 1.1

conax

Chipset Security Assessment
of
Neotion NKE-1

The CAM chipset Neotion NKE-1 has undergone the Conax Security Analysis, determining its potential to make a CAM/STB conform to the security requirements of the Conax Chipset Pairing Integration Kit, version 5.0.

The chipset is assessed to a maximum attainable security level. This represents the best possible evaluation result for a CAM/STB using this particular chipset.

The maximum Conax STB Security level is assessed
Security Evaluation Methodology" version 1.1, using
effective security) to 10.

Security Levels generally decrease over time due to ev
age of this assessment is therefore essential for its
assessment older than 18 months, Conax should be c

The methodology and other classified background of t
in "Security Assessment of the Neotion NKE-1 Backgr
ver.1.1.

Details of the assessment and the tested chipsets are f
complete report "Analysis of the Neotion NKE-1 to su
evaluations", v1.1, dated August 31, 2

The Neotion NKE-1 is assessed to:
Maximum Conax STB Security Level 5

Oslo, October 12, 2010

Torstein Gleditsch
Chief Security Officer
Conax AS

Rune Johansen
Rune Johansen
Manager, Consumer Devices
Conax AS

conax

Conax AS
Kongens gate 8
NO-0153 Oslo • Norway
E: info@conax.com

T: +47 22 40 52 00
F: +47 22 41 70 39
www.conax.com

org.no: NO 963 580 207

Source: Neotion company website

DVB CHIPSETS

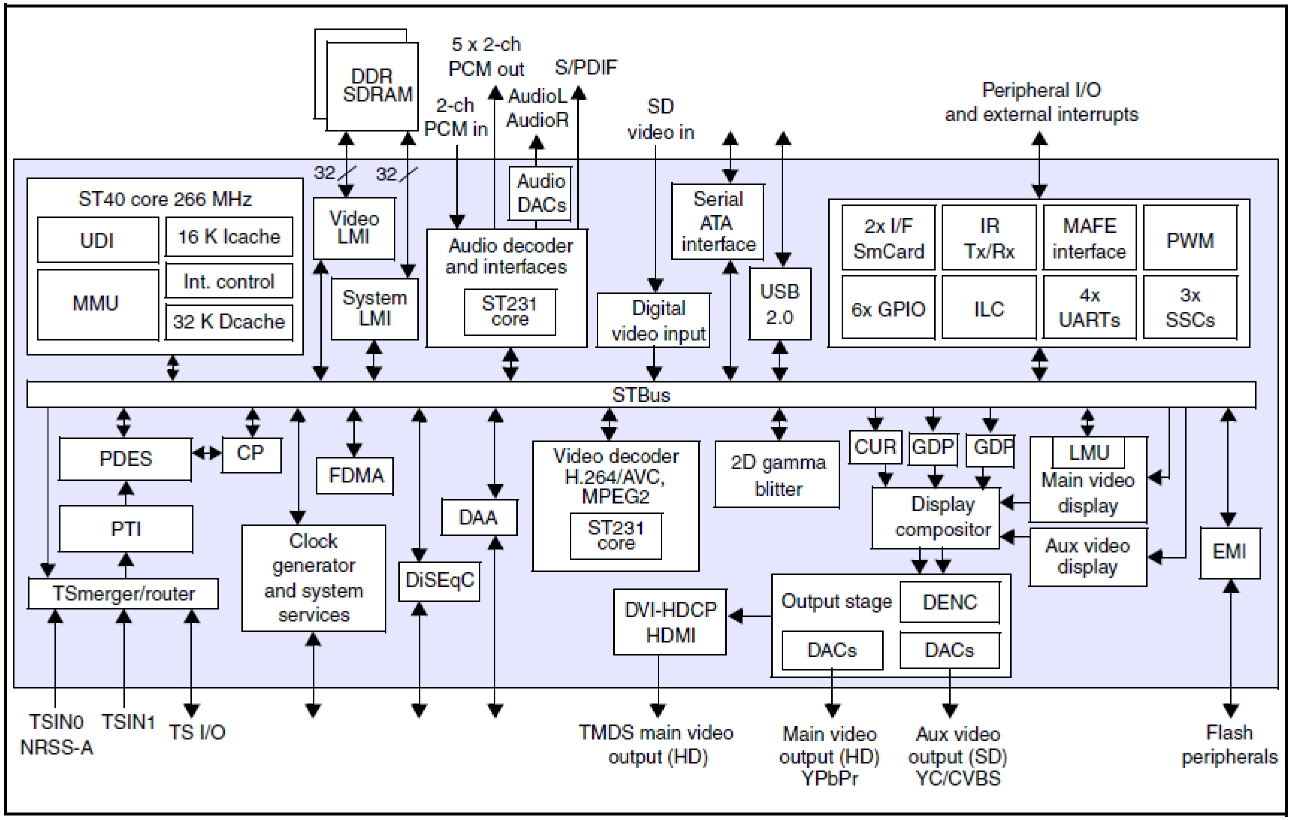
Introduction

- DVB chipsets implement the core functionality related to the handling of MPEG transport streams and A/V data such as:
  - MPEG transport filtering and descrambling (incl. chipset pairing function)
  - audio and video decoding
  - graphics display
  - communication interfaces
  - memory interfaces
- For security and efficiency reasons, they are usually implemented as a single chip (system-on-chip or SoC)
  - Multiple processor cores for various functions

DVB CHIPSETS

STMicroelectronics implementation

- STi7100 single-chip, high-definition STB decoder

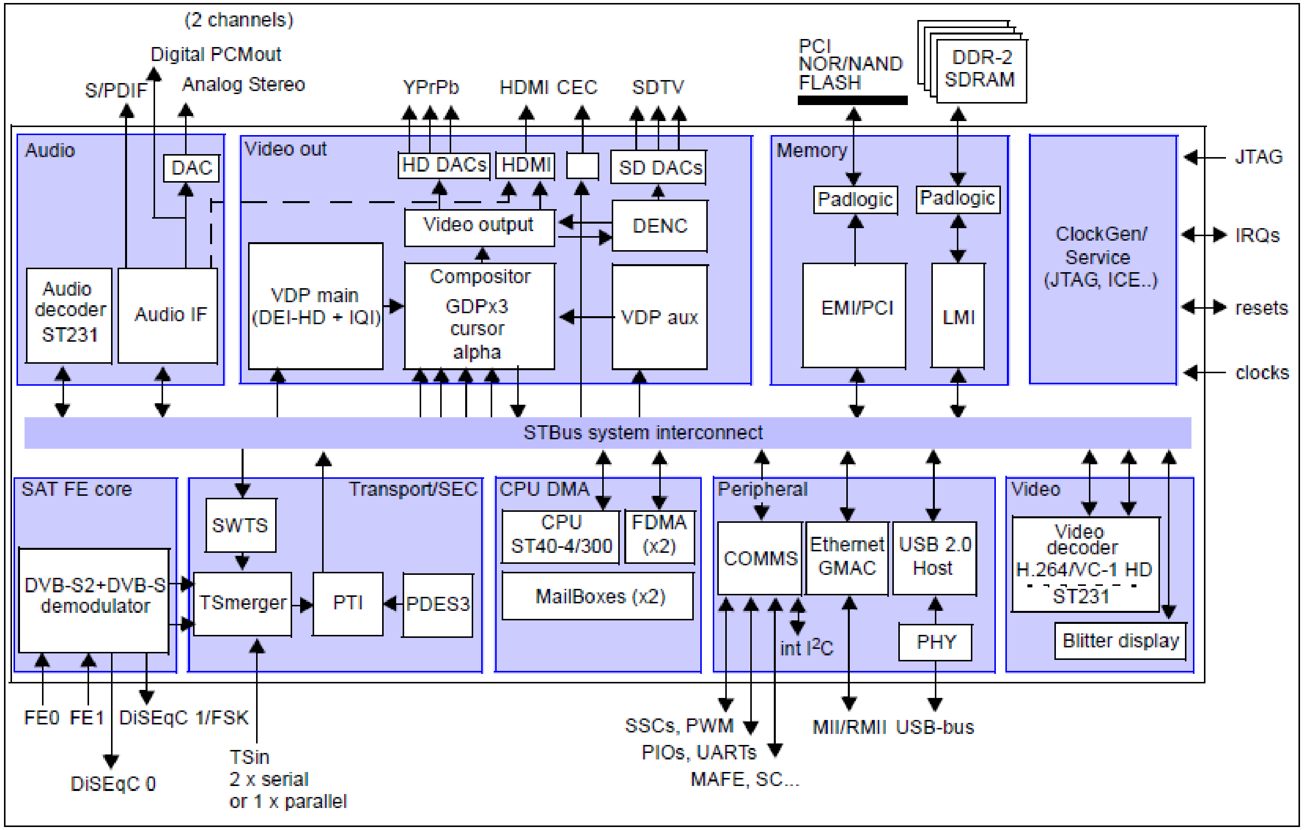


source: st.com

DVB CHIPSETS

STMicroelectronics implementation (2)

- STi7111 single-chip, high-definition STB decoder



source: st.com

DVB CHIPSETS

STMicroelectronics STB H.264 generations

security explorations

GENERATION-1

STi7100
STi7103/Douglas
7109, 5202

90 / 80nm

Mass Production:
2007

ITI5800S
STi7100

ITI5800SX
STi7100

GENERATION-2

STi7105
STi7104/Sequoia

7111, 7141, 7200
5211, 5206

65 / 55 nm

Production Start:
2009

ITI2850ST
STi7111

ITI2849ST
STi7111

Source: ***Multimedia Convergence & ACCI Sector Overview,*** **Philippe Lambinet, STMicroelectronics**

BREAKING DVB CHIPSETS

Security challenges

- Implementation of a chipset pairing function in a proprietary silicon chip makes it far more difficult to reverse engineer and break
  - no target software for the static analysis / reverse engineering or runtime interception
  - undocumented interfaces
  - unknown implementation of the pairing function
  - unknown crypto algorithm and keys (their sizes, byte order, etc.)

BREAKING DVB CHIPSETS

How come ?

- Tedious analytical and reverse-engineering work
- By gathering and gluing together many pieces of information (clues), it was possible not only to discover the operation and implementation of investigated DVB chips, but also find security weaknesses in them
- The tools
  - Without custom reverse engineering tools we would not be able to successfully complete most of our projects
  - This is especially valid for SE-2011-01 project

BREAKING DVB CHIPSETS

Common approach (chips documentation)

- Data briefs available from `st.com` (STi710x, STi7111, STM7710, etc.)
  - Generic chip architectures
  - Processor cores
    - ST40 32-bit superscalar RISC CPU
    - Dual ST231 CPU cores for audio and video decoding
  - Transport subsystem
    - Programmable Transport Interface(PTI)
      - PID filtering, Demultpilexing, Descrambling
    - Transport Stream Merger (TSM) and router
  - FDMA controller
    - PES parsing and start code detection
    - Routing elementary streams to A/V buffers

BREAKING DVB CHIPSETS

Common approach (discovering core device drivers)

- Device drivers implementing Control Words operations
- Static / binary analysis
  - Inspecting libraries and device driver code / symbols
  - Figuring out code dependencies
    - Call and link graphs

GSECHAL device driver (STi7100)

```
gSecHAL_Init
gSecHAL_GetRevision
gSecHAL_SetAlgorithm
gSecHAL_CKCalc
gSecHAL_DecryptSCK
gSecHAL_GetStatus
gSecHAL_CopyTCN
gSecHAL_Reset
...
```

STTKDMA device driver (STi7111)

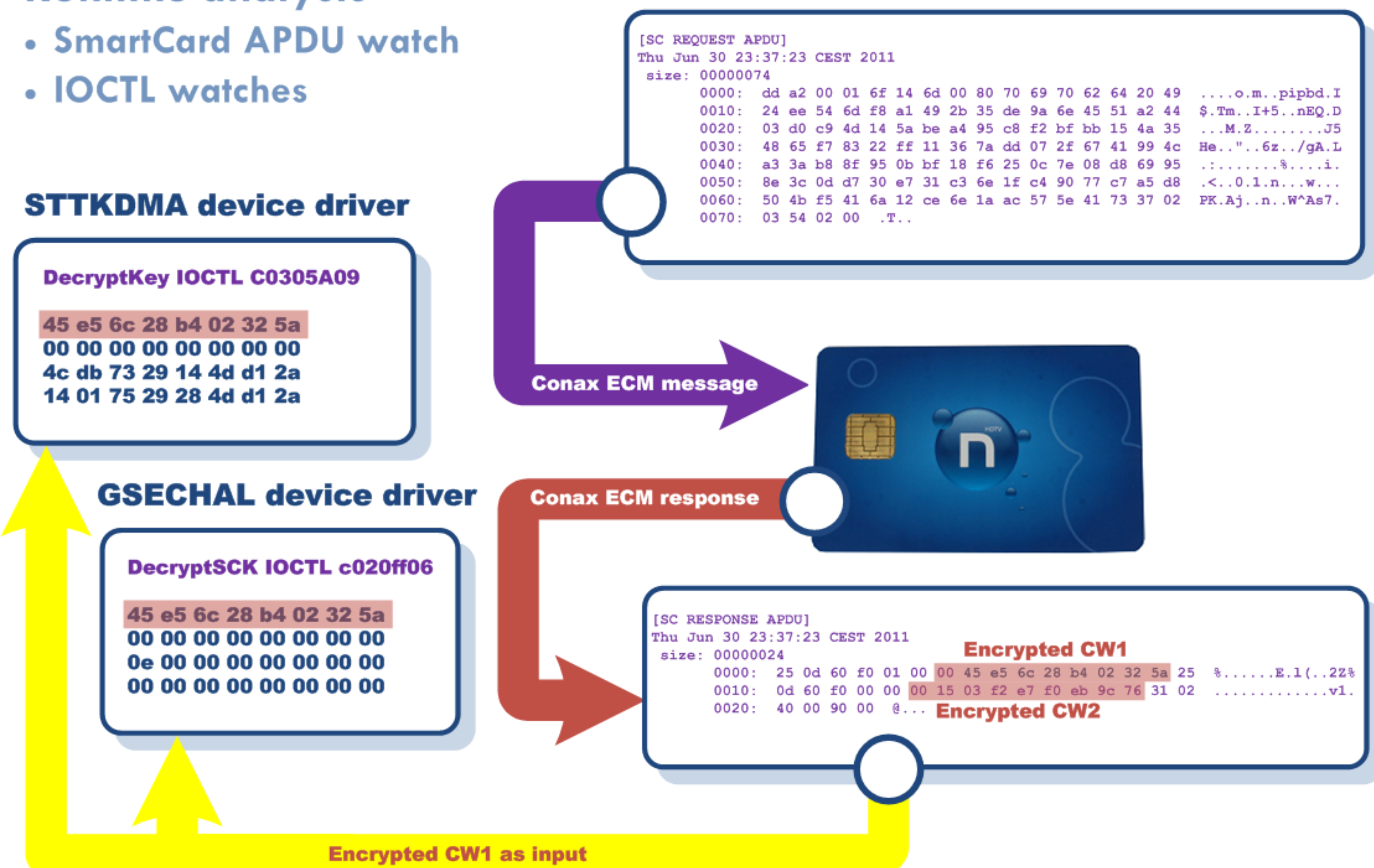
```
STTKDMA_Reset
STTKDMA_DecryptKey
STTKDMA_ReadPublicID
sttkdmaHal_GetNonce
STTKDMA_GetCounter
STTKDMA_Nop
sttkdmaHal_GetSWReg
...
```

BREAKING DVB CHIPSETS

Common approach (pinning down CW API calls)

Runtime analysis

- SmartCard APDU watch
- IOCTL watches



BREAKING DVB CHIPSETS

Helpful CAS system implementation

- Non-HD prepaid satellite TV service (TNK) available along the main Platform 'N'
  - Different set-top-box decoders
    - Technisat, ...
  - Conax CAS smartcards
- A few services available to both SAT TV platforms
  - Shared audio / video streams (same broadcast)
  - Separate conditional access information
    - separate ECM streams

BREAKING DVB CHIPSETS

Helpful CAS system implementation (2)

security explorations

Program Stream Infromation

```
[0] "DIGITAL_TV" "TVN" dvb://13e.3e8.10d7
ServiceDetails: com.adb.dvb.si.AdbSIService, tune_id=4, type=4, version=1, onid=13e, tsid=3e8,
svid=10d7, name=TVN, type=1
PMTService:     com.adb.dvb.si.AdbPMTService, tune_id=4, type=5, version=3, onid=318, tsid=1000,
svid=4311, PCR=512
 PMTElementaryStreams:
   tag 00 type 02 dvb://13e.3e8.10d7.0 PID 0200
   tag 8a type 04 dvb://13e.3e8.10d7.8a PID 028a
   tag 8b type 06 dvb://13e.3e8.10d7.8b PID 028b
   tag 40 type 06 dvb://13e.3
   tag fe type c0 dvb://13e.
   tag fe type c1 dvb://13e.
   tag 01 type 06 dvb://13e.
 Descriptors:
   ...
   tag 0009 UNKNOWN                  len 000f  05 00 e6 4c 10 01 00 13 01 20 14 03 03 2a 00
   ...
   tag 0009 UNKNOWN                  len 0004  0b 00 e5 08
   ...
   tag 0009 UNKNOWN                  len 0004  0b 01 e5 94
   ...
```

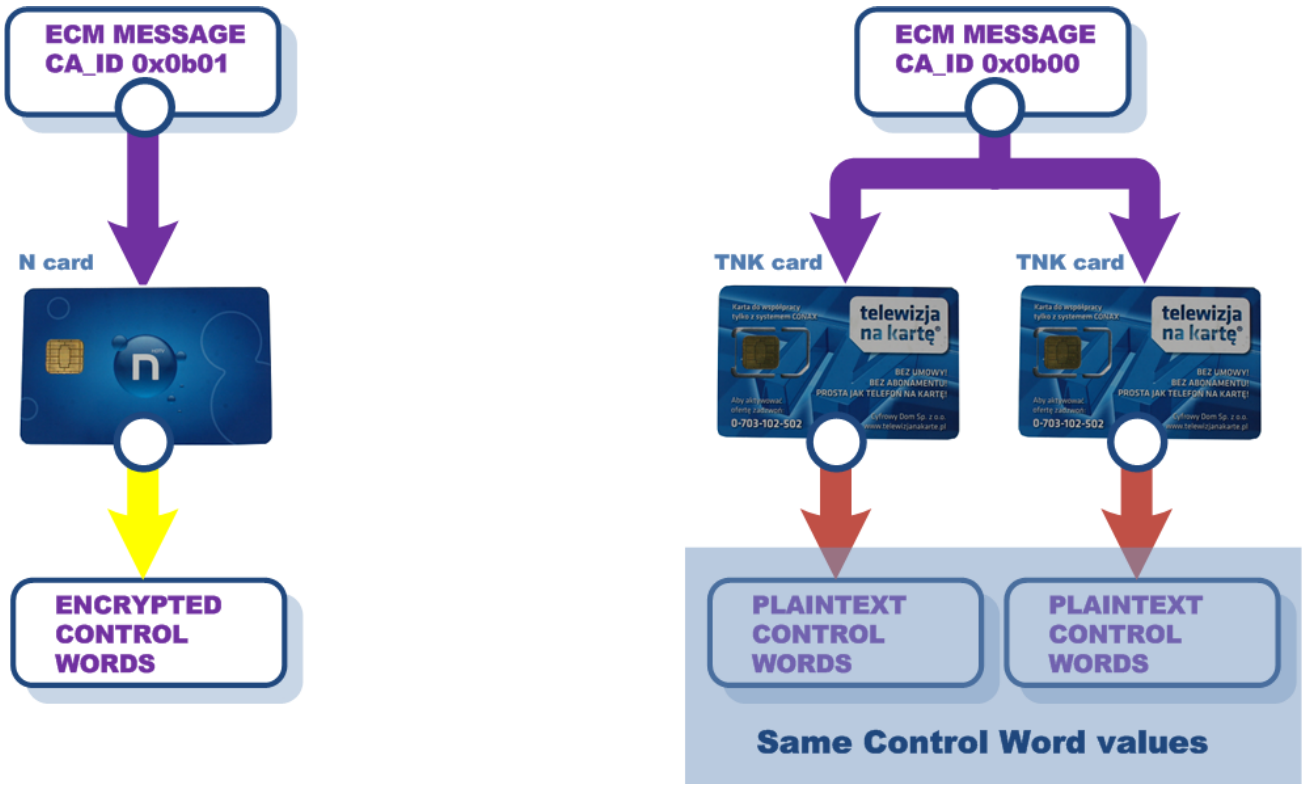
- CA ID 0x0b00 (CONAX CAS)
- PID for ECM stream 0x0508

- CA ID 0x0b01 (CONAX CAS)
- PID for ECM stream 0x0594

BREAKING DVB CHIPSETS

Helpful CAS system implementation (3)

- Parallel Conax CAS **without chipset pairing**
- The plaintext values of encrypted Control Words



BREAKING STi7100 CHIPSET

Device drivers' functionality and operation

- Detailed analysis of GSECHAL's `DecryptSCK` function
  - The meaning of configuration data
    - `SecureMode = 1`
    - `UsingAES = 0`
  - Memory mapped I/O registers
    - CONFIG
    - STATUS
    - COMMAND
    - DATA
  - Implementation of direct chip programming commands
    - `gSecWaitForComplete, gSecDataRead, gSecDataWrite,`

BREAKING STi7100 CHIPSET

Reconstruction of a chip's programming sequences

DecryptSCK IOCTL (pseudocode)

```
If (SecureMode=1, UsingAES=1) {
        gSecDataWrite buf 4
        gSecInstWrite 3|arg<<8
}

If (SecureMode=0, UsingAES=1) {
        d1 -> BASE_ADDR_+0x6100+arg<<4
        d2 -> BASE_ADDR_+0x6104+arg<<4
        d3 -> BASE_ADDR_+0x6108+arg<<4
        d4 -> BASE_ADDR_+0x610c+arg<<4
}

If (SecureMode=1, UsingAES=0) {
        gSecDataWrite buf 2
        gSecInstWrite 3|arg<<8
        gSecInstWrite 3|arg<<8
        gSecSetKeyPtr
}

If (SecureMode=0, UsingAES=0) {
        d1 -> BASE_ADDR_+0x6100+arg<<4
        d2 -> BASE_ADDR_+0x6104+arg<<4
        d3 -> BASE_ADDR_+0x6108+arg<<4
        d4 -> BASE_ADDR_+0x610c+arg<<4
        gSecSetKeyPtr
}
```

DecryptSCK implementation leaks Control Words storage addr (0x6100)

- **Default GSEC chip operation**
- **SecureMode value defined by chip fuses**

BREAKING STi7100 CHIPSET

GSEC keys memory

- Secure storage area for Control Word keys
  - 0x6100 offset from the chip's base addr
  - Unavailable for reading / writing
    - Read operation always returns ZEROs
    - Write operation does not disrupt the descrambling process
- The arguments to the `DecryptSCK` command include the index of the key slot to load with 0x10 bytes (key data)
  - Device driver code makes sure that this index is within the 0x00-0x31 bounds

BREAKING STi7100 CHIPSET

Security vulnerability

- The ability to extract plaintext values of Control Words
  - the chip needs to be programmed manually by issuing commands directly to its I/O mapped registers
  - index of the `DecryptSCK` command needs to be greater than 0x31

plaintext CW mem = 0x6100 + key_idx * 0x10

where key_idx>0x31

BREAKING STi7100 CHIPSET

Security vulnerability (formula)

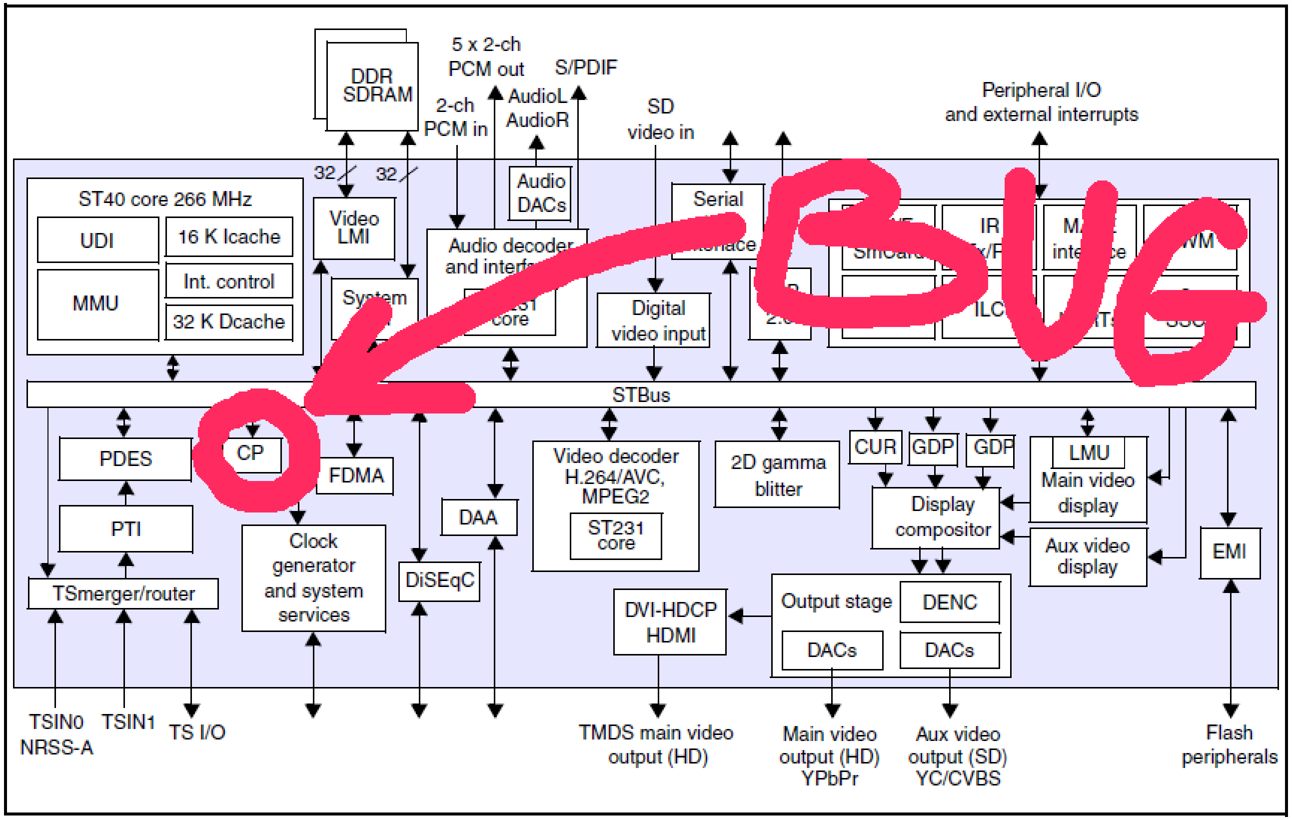
$$Key_{32h} \leftarrow TDES_{dec}(encCW, plainCWPK)$$
$$Key_{32h} \equiv plainCW$$

$Where:$

$Key_{32h} = key\ slot\ \#32h\ (accessible\ chip\ location)$
$encCW = encrypted\ Control\ Word\ value\ (known\ value)$
$plainCWPK = plaintext\ Pairing\ Key\ value\ (unknown\ value)$

BREAKING STi7100 CHIPSET

Security vulnerability (SoC location)



original image: st.com

BREAKING STi7111 CHIPSET

Introduction

- Different design brings more challenges
  - STTKDMA chip component
  - One `DecryptKey` IOCTL for key related operations
    - Control Words
    - Pairing Key (CWPK)
    - AES keys
- Easier reverse engineering
  - Modular architecture of ITI2850ST/ ITI2849ST set-top-boxes' OS distribution
    - Many dedicated user level libraries
      - Conax CA, Crypto operations, NAND encryption, STB configuration, …
    - Text XML configuration files
      - Conax CA client settings
  - Support for crypto DMA operations
  - Kernel symbols via `/proc/kallsyms` (680KB+)

BREAKING STi7111 CHIPSET

Dedicated key memories

- Separate memory mapped chip regions for AES and Control Word keys
  - Deduced with the help of kernel and user level library symbols
  - Code / data symbols associated with accesses to chip's I/O memory

DecryptKEY FUNCTIONALITY

cpcw_keys, descrambling keys (set_cleartext_descramblingkey)
3100 offset, indices 0x00-0x31, key size 0x10

CRYPTO DMA FUNCTIONALITY

sttkdma keys, cdma_dev_keys, cpcwKeys (STDRMCRYPTO_AES_LoadKeySlot)
3420 offset, indices 0x00-0x07, key size 0x10

BREAKING STi7111 CHIPSET

CCORE library

- The library used for crypto DMA operations
- Reverse engineering the meaning of CCORE library/ chips configuration bits and input / output arguments
  - Manual analysis of data propagation
    - `libstd_drv_ccore.so` API -> STTKDMA device driver API -> chip's configuration registers
      - DMA CONFIG
      - TKD CONFIG
  - Analysis of STTKDMA code writing to configuration registers
    - `resetAES_NOT_TDES`
- Custom AES / TDES Java subroutines
  - Verification of CCORE results

BREAKING STi7111 CHIPSET

The existence of a chip specific SCK key

- Initial hints in STi7100 GSECHAL device driver
  - `DecryptSCK` command
- Confirmed at a time of the analysis of a set-top-box boot loader code (SH4 emulator)
  - The use of SCK key to decrypt the boot loader code
    - No initialization of the usual key registers
    - Different chip configuration bits
- Used by a device driver from a software upgrade OS distribution
  - The use of SCK key for NAND encryption
    - `parm=nand_crypt_use_sck_key:Use SCK key instead of default ADB key for NAND encryption`

BREAKING STi7111 CHIPSET

Firmware data / code for STi7111 chip

- Hints in `sttkdma_core_user.ko` module
  - Code symbols
    - `st_tkdma_loader`
    - `st_tkdma_loader_checksum`
  - Data symbols
    - `tkdma_fw_address_1`
    - `tkdma_fw_address_2`
  - Writing data / code to STi7111 chip's I/O space
    - 5944 code bytes
    - 1156 data bytes
- Firmware code implementing unknown processor instructions

BREAKING STi7111 CHIPSET

SLIM Core processor

- IST FP6 PROSYD EU project (http://www.prosyd.org)
- Paragraph 2.3 of Deliverable D1.4/1 gives some information about the SLIM Core Processor
  - a collaboration between ST UK and OneSpin after the spin-off from Infineon
  - lightweight processor with 27 instructions and a 4-stage pipeline
  - processor special features: a coprocessor interface; circular buffer operation; a STOP instruction
  - instructions opcode names: ADD - BRA - CPI - JAB - LD - LDF - NOP - RPT - STI -STF - STOP – SUBC.

BREAKING STi7111 CHIPSET

JMP instruction format

- OpenDuckbox project (http://gitorious.org/open-duckbox-project-sh4)
  - GNU source code for SLIM Core Generic driver
  - `slim_boot_core` function leaks information about the format of one Slim Core instruction (JMP)
    - memory addressing (by word number)
    - instruction opcode width

```
// Init imem so every instruction is a jump to itself
  for (n = 0; n < core->imem_size / 4; n++)
       SLIM_IMEM(core, n) = 0x00d00010 | (n & 0xf) |
                            ((n & 0xff0) << 4);
```

BREAKING STi7111 CHIPSET

Finding patterns in SLIM Core code

Result of GetPublicID command

0x4020	0x4024	0x4028	0x402c
CHIP ID	0x00000000	0x00000000	0x00000000

SLIM Core firmware sequence

```
01a1  0x00a5008b   ld r5,[r0,008b] // 0x5e2c
01a2  0x00a9001f   ld r9,[r0,001f] // 0x407c
01a3  0x00b05008   st r5,[r0,0008] // 0x4020
01a4  0x00b00009   st r0,[r0,0009] // 0x4024
01a5  0x00b0000a   st r0,[r0,000a] // 0x4028
01a6  0x00b0000b   st r0,[r0,000b] // 0x402c
01a7  0x00d01c1a   jmp l_01ca
```

Discovery of STORE instruction format

- Firmware pattern matching format of GetPublicID result
- Confirmed by changing ST R5 with ST R0 instruction

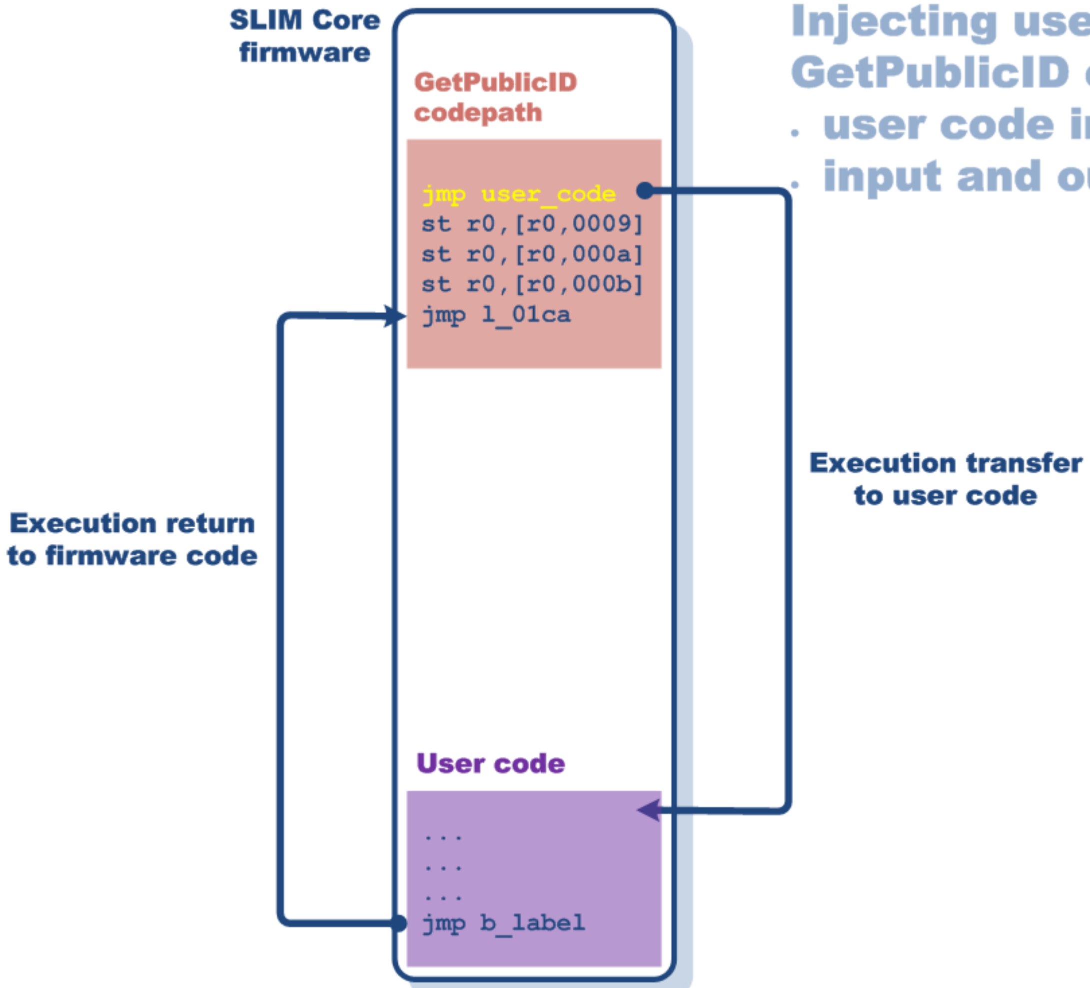
BREAKING STi7111 CHIPSET

Reverse engineering SLIM Core instructions

- Discovery of SLIM Core instruction opcodes
  - Exploited the ability to change the operation of SLIM Core firmware in runtime
    - Overwriting chip's memory loaded with firmware code
    - **should secure crypto chip allow for it ?**
  - Replacement of an arbitrary instruction from the code path of `GetPublicID` function
  - Analysis of the instruction's execution effect to memory and registers
    - Discovery of load / store instructions format

BREAKING STi7111 CHIPSET

Reverse engineering SLIM Core instructions (2)



Injecting user code into GetPublicID codepath

- user code in unused space
- input and output arguments

BREAKING STi7111 CHIPSET

Reverse engineering SLIM Core instructions (3)

- JMP and LOAD/STORE instructions sufficient to discover the meaning of all other instructions
  - **JMP** from firmware to user's code path
    - **STORE** the contents of registers (firmware context)
      - **LOAD** user's environment (contents of registers)
      - **EXECUTE** unknown SLIM Core instruction opcode
      - **STORE** user's environment (contents of registers)
    - **LOAD** the contents of registers (firmware context)
  - **JMP** back to firmware code path
- The need to properly handle conditional jumps

BREAKING STi7111 CHIPSET

Reverse engineering SLIM Core instructions (4)

- One instruction opcode at a time
  - LOAD / STORE instructions
  - MOV instructions
  - CMP instructions
  - Conditional branching instructions
  - Computational instructions
  - Other instructions (bit extraction, manipulation)
  - RPT instruction
- Scope limited to unknown opcodes from firmware code

BREAKING STi7111 CHIPSET

Reverse engineering SLIM Core instructions (5)

Visible code patterns

MOV instruction patterns

```
0003   0x00e30374   mov r3,#0374
0004   ....
0005   0x00e4ffff   mov r4,#ffff
0006   0x00e3ffff   mov r3,#ffff
0007   ....
0008   0x00e30001   mov r3,#0001
```

MOV instruction confirmed by the result of STORE

CMP / conditional jump instruction patterns

```
0014   0x00981026   je l_0026
0015   0x00c03002   cmp r3,#02
0016   0x00981028   je l_0028
0017   0x00c03006   cmp r3,#06
0018   0x0098102a   je l_002a
0019   0x00c0300b   cmp r3,#0b
001a   0x0098102c   je l_002c
001b   0x00c0300f   cmp r3,#0f
001c   0x0098102e   je l_002e
001d   0x00c03003   cmp r3,#03
001e   0x00981030   je l_0030
001f   0x00c03007   cmp r3,#07
```

CMP values correspond to device driver commands

BREAKING STi7111 CHIPSET

SLIM Core disassembler

- Final disassembly dump of Slim Core firmware code
  - 1400+ instructions disassembled
  - ~11 instruction opcodes not recognized
    - Not relevant from the analysis point of view
- Sufficient data for firmware analysis
  - Discovery of separate dispatching for DMA and all TKD operations
    - Semi-threads (context-switching)
  - Discovery of a key initialization subroutine

BREAKING STi7111 CHIPSET

Tracing SLIM Core firmware

- The goal was to locate SLIM Core instruction sequences implementing `DecryptKey` functionality
- Tracer implementation
  - SLIM core part
    - Custom code on the `GetPublicID` function path
      - Binary instrumented instruction copied from a currently traced code location
      - SLIM Core instruction executed in the original registers context
      - Heavy use of the SLIM core disassembler
  - Java part
    - Logging
    - SLIM Core syncing and control code

BREAKING STi7111 CHIPSET

Tracing SLIM Core firmware (output log)

```
starting logger at 80
break at: 0x00000086
 r0  00000000 *r1  00000001 *r2  00000100  r3  00000000
*r4  00000011 *r5  00000031 *r6  00001103 *r7  00000005
*r8  00000006 *r9  31ff0001  r10 00000000 *r11 00000001
 r12 00000000 *r13 0000024e *r14 000000d0  IP  00000086
0086  0x00e10001   mov r1,#0001

SCDC stopped
0086  0x00e10001   mov r1,#0001
break at: 0x00000087
 r0  00000000  r1  00000001  r2  00000100  r3  00000000
 r4  00000011  r5  00000031  r6  00001103  r7  00000005
 r8  00000006  r9  31ff0001  r10 00000000  r11 00000001
 r12 00000000  r13 0000024e  r14 000000d0  IP  00000087

0087  0x00a20048   ld r2,[r0,0048] // 0x4120
break at: 0x00000088
 r0  00000000  r1  00000001 *r2  00000001  r3  00000000
 r4  00000011  r5  00000031  r6  00001103  r7  00000005
 r8  00000006  r9  31ff0001  r10 00000000  r11 00000001
 r12 00000000  r13 0000024e  r14 000000d0  IP  00000088

0088  0x00722c21   bitval r2,r2,#0002
0089  0x00881091   jz l_0091
break at: 0x00000091
 r0  00000000  r1  00000001 *r2  00000000  r3  00000000
 r4  00000011  r5  00000031  r6  00001103  r7  00000005
 r8  00000006  r9  31ff0001  r10 00000000  r11 00000001
 r12 00000000  r13 0000024e  r14 000000d0  IP  00000091
```

BREAKING STi7111 CHIPSET

Tracing SLIM Core firmware (DecryptKey code)

DecryptKey implementation

```
0206 0x00fa4000 copTDES
0207 0x000f093c mov r15,r9
0208 0x008e1208 wait1
0209 0x00af0008 ld r15,[r0,0008]
020a 0x00af0009 ld r15,[r0,0009]
020b 0x00af000a ld r15,[r0,000a]
020c 0x00af000b ld r15,[r0,000b]
020d 0x008e120d wait1
020e 0x00500a00 tst r10,r10
020f 0x00881215 jz l_0215
0210 0x00b0f008 st r15,[r0,0008]
0211 0x00b0f009 st r15,[r0,0009]
0212 0x00b0f00a st r15,[r0,000a]
0213 0x00b0f00b st r15,[r0,000b]
0214 0x00d02117 jmp l_0217
0215 0x00d00004 rpt 4
0216 0x00000f3c mov r0,r15
0217 0x00d0211a jmp l_021a
```

- 0206: put chip into TDES mode
- 0207: load TKD command
- 0208: wait
- 0209–020c: load encrypted CW
- 020d: wait
- 0210–0213: optionally store result

BREAKING STi7111 CHIPSET

Internal crypto core (TKD) commands

- Static analysis of SLIM Core firmware disassembly
  - Discovery of internal chip commands

```
00ff8101 stk cmd 2                setCWPK
01ff8101 stk cmd 1
02ff8101 stk cmd 0x20
03ff0001 stk cmd 0x10

10ff8001 stk cmd idx<<8 | 0x80
10ff0101 stk cmd idx<<8 | 0x04

20ff0001 stk cmd idx<<8 | 0x03  set_protected_descramblingkey
20ff0010 stk cmd idx<<8 | 0x06

04000001 stk cmd 0x11

ffff0401 stk cmd 0x12
80ff0203 stk cmd 0x21
81ff0203 stk cmd 0x22
82ff0203 stk cmd 0x23
83ff0203 stk cmd 0x24
```

Set CWPK	**0x00ff8101 CMD**
Set encrypted CW idx 0x00	**0x20ff0001 CMD**
Set encrypted CW idx 0x01	**0x21ff0001 CMD**
Set encrypted CW idx 0x02	**0x22ff0001 CMD**
Set encrypted CW idx 0x03	**0x23ff0001 CMD**
...	
Set encrypted CW idx 0x31	**0x51ff0001 CMD**

BREAKING STi7111 CHIPSET

TKD inspector

- Discovery of the meaning of TKD commands by the means of executing special SLIM Core instruction sequence

```
public static final int tkd_code[] = {
        0x00e61234,    //mov r6,#1234 TKD_CMD_HI
        0x00e55678,    //mov r5,#5678 TKD_CMD_LO
        0x00e00000,    //mov r0,#0000
        0x00756210,    //mov r5,(r6<<16)|r5
        0x00e00000,    //mov r0,#0000
        0x00fa4000,    //COPINS
        0x00e00000,    //mov r0,#0000
        0x000f053c,    //mov r15,r5
        0x00e00000,    //mov r0,#0000
        0x008e1abc,    //WAITINS
        0x00e00000,    //mov r0,#0000
        0x00af0050,    //ld r15,[r0,0050]
        0x00af0051,    //ld r15,[r0,0051]
        0x00af0052,    //ld r15,[r0,0052]
        0x00af0053,    //ld r15,[r0,0053]
        0x00e00000,    //mov r0,#0000
        0x008e1abc,    //WAITINS
        0x00e00000,    //mov r0,#0000
        0x00b0f054,    //st r15,[r0,0054]
        0x00b0f055,    //st r15,[r0,0055]
        0x00b0f056,    //st r15,[r0,0056]
        0x00b0f057,    //st r15,[r0,0057]
        0x00e00000     //mov r0,#0000
};
```

TKD command to test

Chip mode instruction (AES/TDES)

Wait ins (different for AES/TDES mode)

Input data (key)

Wait ins (different for AES/TDES mode)

Output data (key)

BREAKING STi7111 CHIPSET

TKD inspector (playing with the commands)

```
test> input "b6 04 78 c3 0f 26 a3 06 d5 20 10 0f c0 93 4f f3"
test> ed 0x01ff0000 0x00fa4000 0x008e1abc
tkcmd 01ff0000

[running SLIM code]
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

test> input "e0 c9 cd 4e 2f bd 52 a0 e0 c9 cd 4e 2f bd 52 a0"
test> ed 0x15ff0101 0x00fa4000 0x008e1abc
tkcmd 15ff0101

[running SLIM code]
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

test> keys
[00] 6d 31 99 ca f8 fa e5 51 fc 56 22 11 c2 33 05 8a
[01] 6d 31 99 ca f8 fa e5 51 fc 56 22 11 c2 33 05 8a
[02] 6d 31 99 ca f8 fa e5 51 fc 56 22 11 c2 33 05 8a
[03] f8 25 41 20 00 00 00 00 00 00 00 00 00 43 11 71
[04] 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[05] 00 85 55 26 86 09 29 54 00 85 55 26 86 09 29 54
[06] 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[07] d4 c8 94 af 84 84 5c de 17 82 f7 73 1e c3 2f e7
DMA CONFIG: 20 08 00 00
TKD CONFIG: 00 00 00 00
INPUT: e0 c9 cd 4e 2f bd 52 a0 e0 c9 cd 4e 2f bd 52 a0
```

BREAKING STi7111 CHIPSET

TKD commands format

31 24 16 8 0

TARGET	SOURCE	KEY	CONFIG

TARGET:
- target, where the result of the operation should be stored
- a key slot number or 0xff for chip registers

SOURCE:
- source, from which data for the operation should be fetched
- a key slot number or 0xff for chip registers,

KEY:
- key slot number, which holds the key used for the crypto operation
- value 0x00 usually identifies SCK key (unique key for each chip)

CONFIG:
- configuration bits
- bit 0 Usually denotes encryption (0) or decryption (1) operation

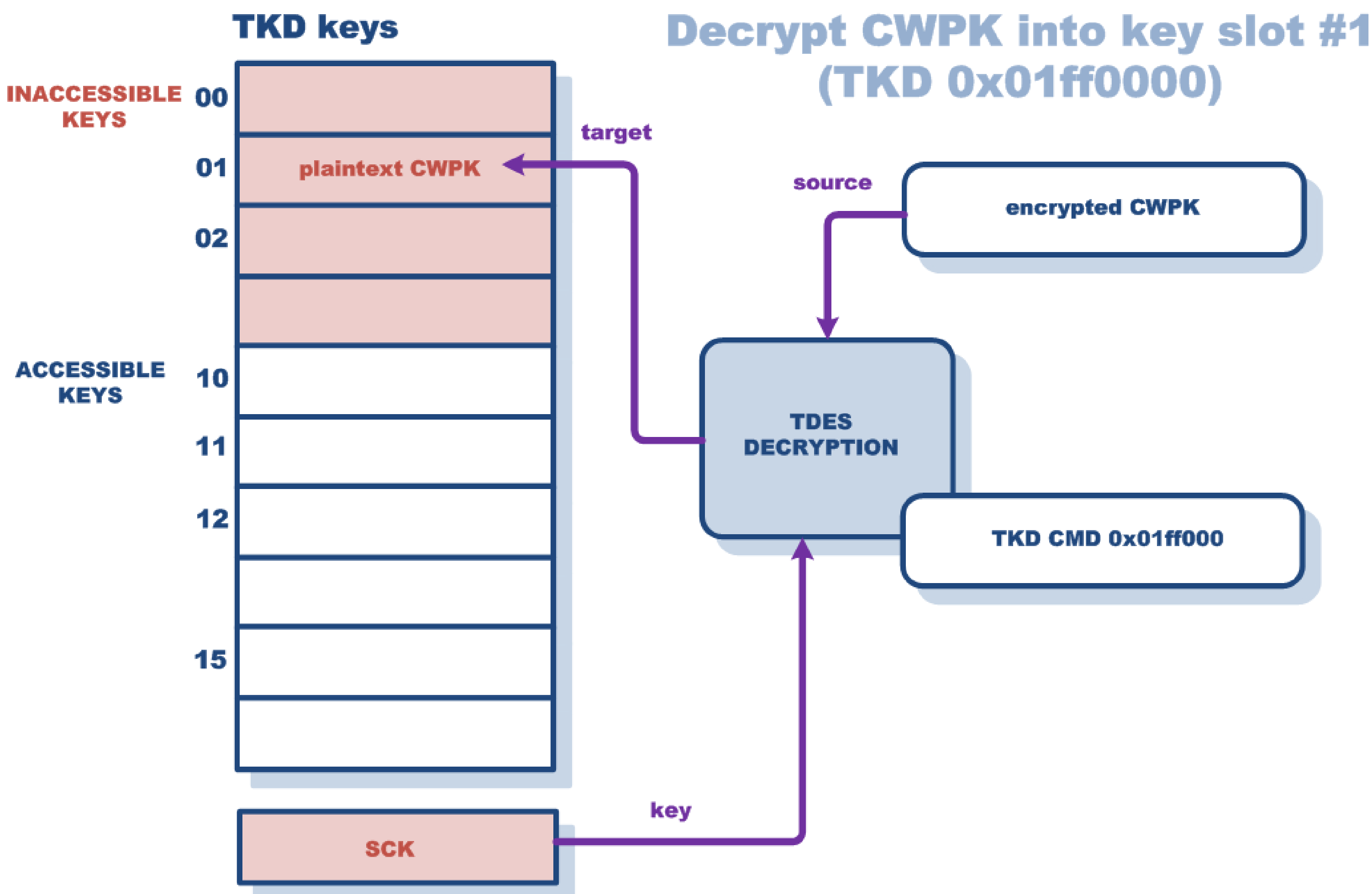
BREAKING STi7111 CHIPSET

TKD commands format (key operations explained)

- TKD CMD 0x00ff0000
  - Setting encrypted Control Word Pairing Key (CWPK)
  - Interpreted as decryption (always) of register input (0xff) with SCK key (0x00) and storing the result at a key slot 0x00
- TKD CMD 0x20ff0001
  - Setting encrypted Control Word
  - Interpreted as decryption (0x01) of register input (0xff) with SCK key (0x00) and storing the result at a key slot 0x20

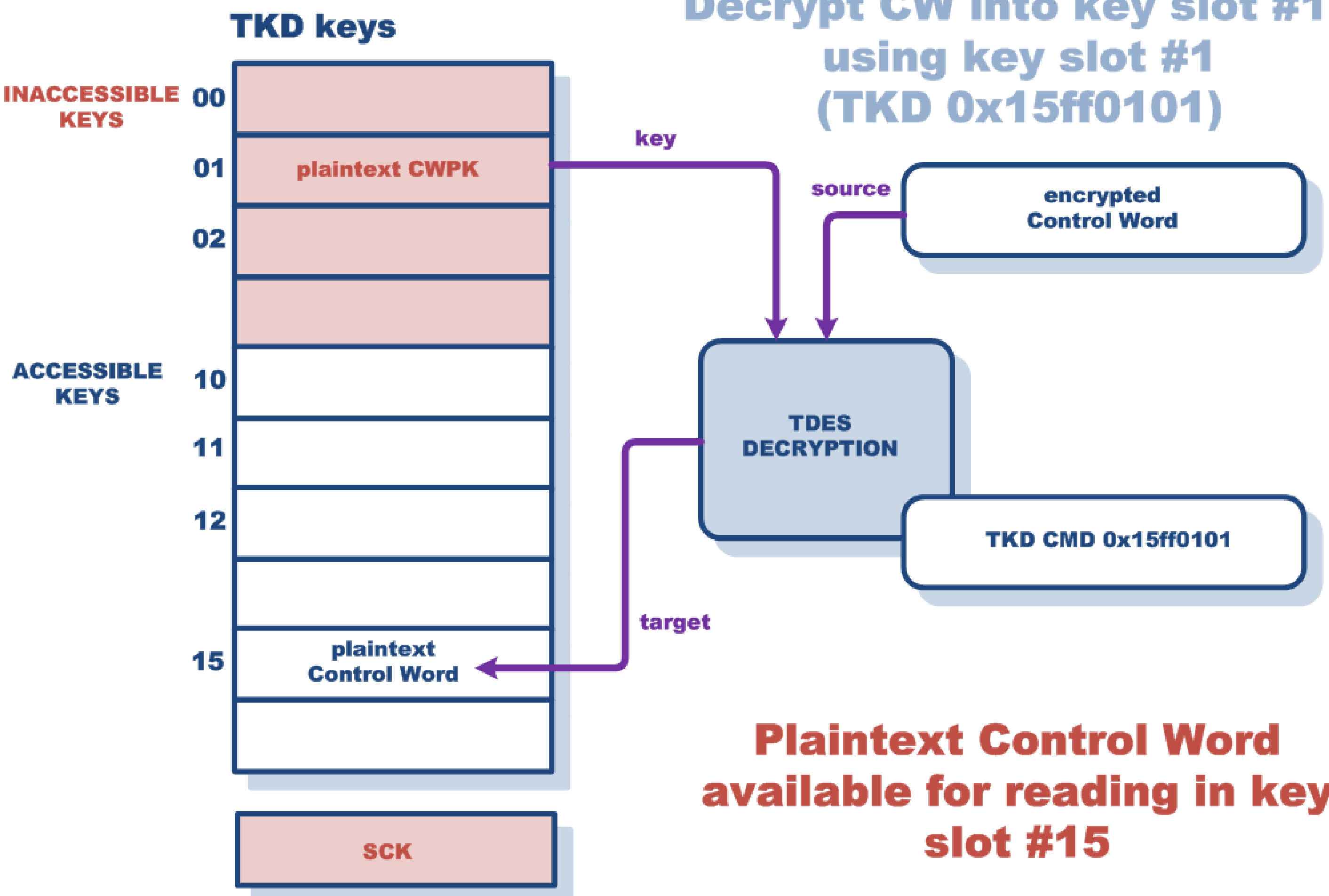
BREAKING STi7111 CHIPSET

Security vulnerability #1 (step 1)



BREAKING STi7111 CHIPSET

Security vulnerability #1 (step 2)



Decrypt CW into key slot #15 using key slot #1 (TKD 0x15ff0101)

Plaintext Control Word available for reading in key slot #15

BREAKING STi7111 CHIPSET

Security vulnerability #1 (formula)

$$Key_1 \leftarrow TDES_{dec}(encCWPK, SCK)$$

$$Key_{15h} \leftarrow TDES_{dec}(encCW, Key_1)$$

$$Key_1 \equiv plainCWPK$$

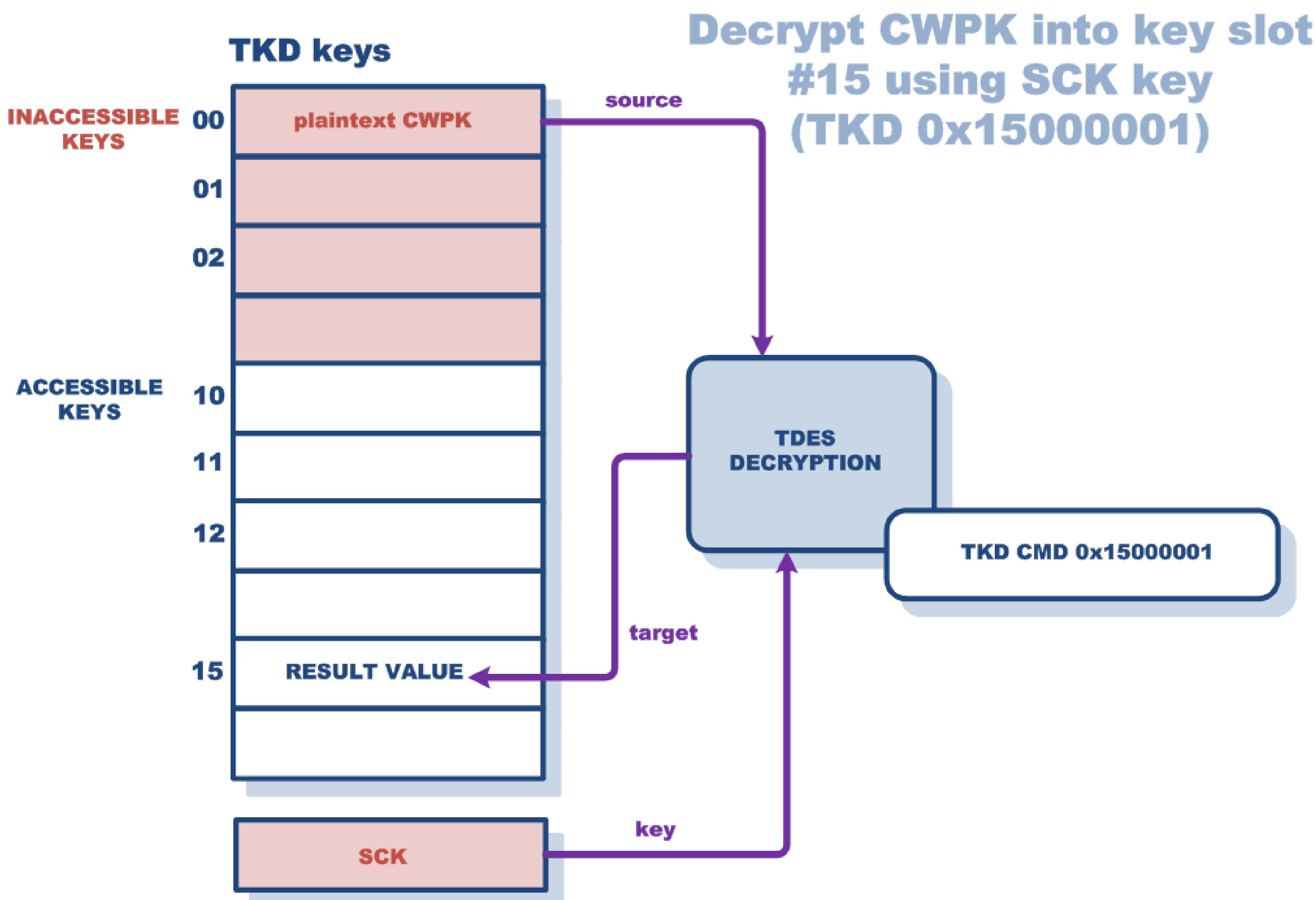
$$Key_{15h} \equiv plainCW$$

Where:

- $Key_1 =$ *key slot #01h (inaccessible chip location)*
- $Key_{15h} =$ *key slot #15h (accessible chip location)*
- $encCW =$ *encrypted Control Word value (known value)*
- $encCWPK =$ *encrypted Control Word Pairing Key value (known value)*
- $SCK =$ *chip specific key (unknown value)*

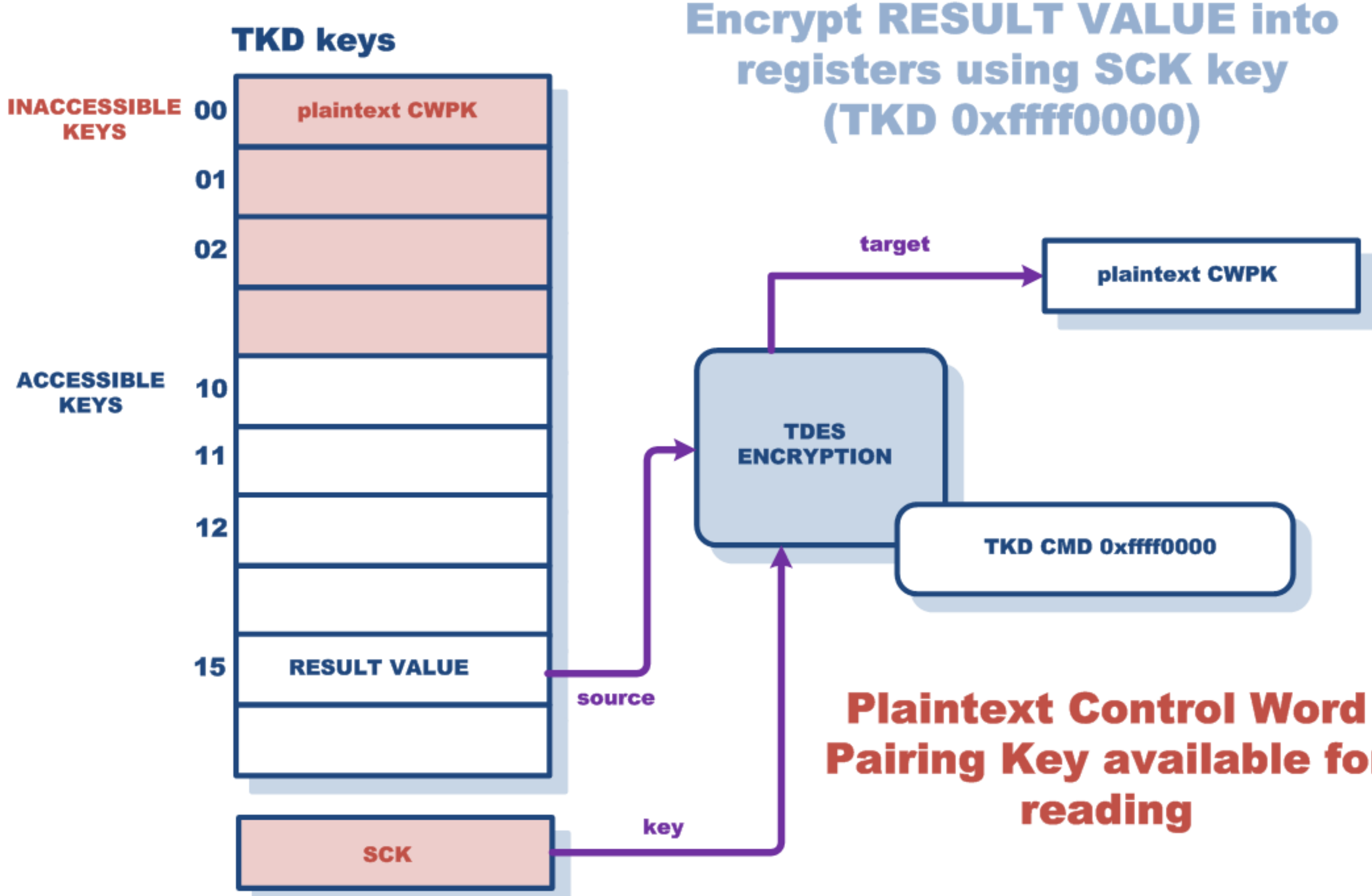
BREAKING STi7111 CHIPSET

Security vulnerability #2 (step 1)



BREAKING STi7111 CHIPSET

Security vulnerability #2 (step 2)



Encrypt RESULT VALUE into registers using SCK key (TKD 0xffff0000)

Plaintext Control Word Pairing Key available for reading

BREAKING STi7111 CHIPSET

Security vulnerability #2 (formula)

$$Key_{15h} \leftarrow TDES_{dec}(plainCWPK, SCK)$$
$$plainCWPK \leftarrow TDES_{enc}(Key_{15h}, SCK)$$

Where:

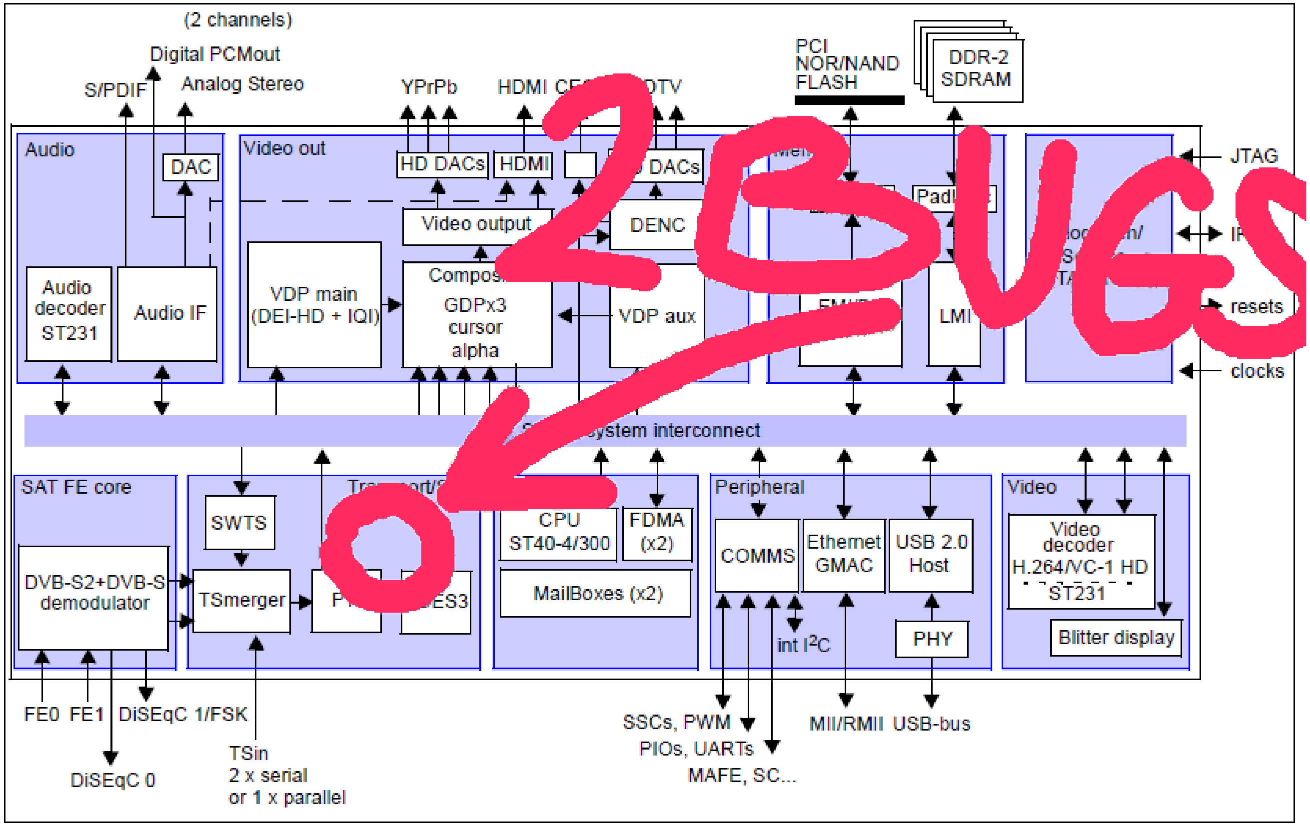
Key_{15h} = *key slot #15h (accessible chip location)*

SCK = *chip specific key (unknown value)*

$plainCWPK$ = *plaintext Pairing Key value (unknown value)*

BREAKING STi7111 CHIPSET

STi7111 security vulnerabilities (SoC location)



original image: st.com

BREAKING STi7111 CHIPSET

Control Word Pairing Key

- Issue 18 makes use of the encrypted value of the chipset pairing key (CWPK)
- CWPK key sent by the operator at the time of activating user's subscription
  - Encrypted CWPK key bytes returned by the Conax card in response to EMM message
- For ITI2850ST and ITI2849ST set-top-boxes encrypted CWPK key encrypted again and cached in a file
  - `/mnt/flash/secure/7/0`
  - `libstd_cai_client_conax7.so` API for decryption

PROOF OF CONCEPT CODE

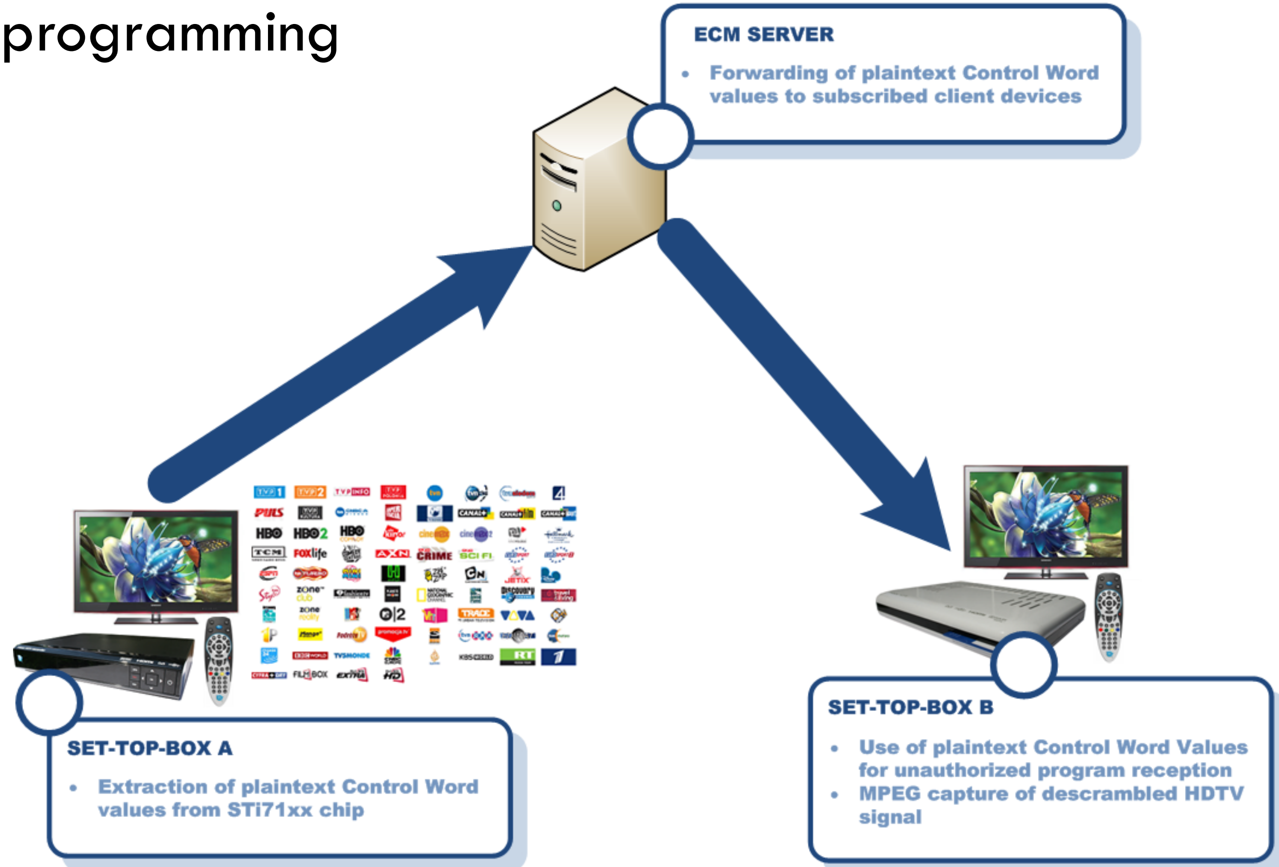
Implemented functionality

- Access to information about cryptographic keys
  - Plaintext Control Words (STi7100)
  - Plaintext Control Words and plaintext CWPK (STi7111)
- Control Words sharing via network between arbitrary decoders protected with Conax conditional access method and chipset pairing
- Video on Demand ECM decryption and sharing of programming protected with Conax conditional access method with chipset pairing

PROOF OF CONCEPT CODE

Control Words sharing (aka. signal theft)

- Sharing of the crypto keys used to descramble digital satellite TV programming



PROOF OF CONCEPT CODE

Push VOD sharing

- Sharing of the crypto keys used to descramble VOD movies
  - Obtaining Control Word for arbitrary movie during the rental period
    - VOD movies rented for 48 hours period
    - Encrypted MPEG data pushed into set-top-boxes
    - ECM messages accompanying movies files
  - Sharing Control Words after the rental period
  - Using plaintext CW values to descramble the movie
    - The use of key memory beyond index 0x32 on STi7100
    - The use of CWPK to reencrypt Control Word on STI7111

SUMMARY

STi7100 / STi7111 security vulnerabilities

- How come the issues were not discovered before the market release ?
  - STMicroelectronics a major silicon vendor
    - #1 in Europe , #7 in the world (source: Wikipedia)
  - Conax security evaluation of STB / CAM / DVB chipset solutions
    - Final scoring of STi7100/STI7111 not disclosed to Security Explorations

SUMMARY

Vulnerabilities Impact

- No information from STMicroelectronics (DVB chipsets vendor) in response to the impact inquiry questions
  - ***All your inquiries, as listed below, are pointing towards confidential information and as such can not be disclosed by ST to you or to others.***

    Jan-17-2011, STMicroelectronics in an email to Security Explorations
- Impact estimation upon publicly available data

SUMMARY

Vulnerabilities Impact (2)

- Cumulative MPEG-2 & MPEG-4 Shipments in 2008
  - **541 millions of units**
    - Set-top-boxes, digital television sets, DVD / Bluray players
- STMicroelectronics #1 in H.264 market (68% of market share in 2008)
- Customers from Europe, Middle East and Africa, Asia-Pacific and the Americas
  - DishTV (India)
  - DirectTV (USA)
  - Platforma N, Cyfrowy Polsat (Poland)
  - BSkyB (UK)
  - …

Source: Multimedia, Philippe Lambinet, STMicroelectronics

SUMMARY

Final Words

- First successful attack against the implementation of a Conax conditional access system with chipset pairing
  - Pay TV piracy possible in the environment of hacked digital satellite TV set-top-boxes
  - Security of dedicated DVB chipsets broken
- Security based on a complex, secret functionality embedded in a silicon is a dangerous concept
  - Security through Obscurity ?
- The need to improve security and evaluation processes by silicon and CAS vendors
- The need to tighten set-top-boxes security relying on vulnerable DVB chipsets

FINAL

Q & A

THANK YOU

contact@security-explorations.com