

# Reverse engineering tools for ST DVB chipsets

SRP-2018-01-LEAFLET

## I. Description of the reverse engineering tools

| 1. SlimCORE disassembler |   |
|--------------------------|---|
| Description              | The tool to disassemble SlimCORE processor instruction streams from various firmwares used by STi7111 DVB chipsets  |
| Features                 | <ul style="list-style-type: none"> <li>▪ SlimCORE instruction stream disassembly from a device driver file or input files corresponding to firmware code / data sections</li> <li>▪ extraction of SlimCORE firmware data / code sections from a device driver file to output files</li> <li>▪ statistics information regarding the usage of SlimCORE instructions (i.e. unknown, recognized instructions)</li> </ul>  |
| Impact                   | <ul style="list-style-type: none"> <li>▪ instrumental to analyze the operation of SlimCORE processor firmware and to discover vulnerabilities in STi7111 DVB chipsets (Issues 17-19)</li> <li>▪ discovery of separate dispatching for DMA and all TKD operations</li> <li>▪ discovery of a key initialization subroutine</li> </ul>   |
| Public reference         | <ul style="list-style-type: none"> <li>▪ HITB talk #2 (slide 50)</li> </ul>   |
| Deliverables             | <ul style="list-style-type: none"> <li>▪ source code</li> <li>▪ a technical document (APPENDIX A) presenting a brief analysis of SlimCORE processor instruction set and operation of the firmware found in ADB STBs (models ITI-2849ST and ITI-2850ST, STTKDMA-REL 3.1.6, 3.5.0 and 3.9.2) that was used as a base material during reverse engineering of TKD core operation</li> <li>▪ original reverse engineering annotations (APPENDIX B) for SlimCORE firmware 3.1.6, 3.5.0 and 3.9.2 (for use with a SlimCORE disassembler to obtain annotated / commented code)</li> </ul> |
| 2. SlimCORE tracer       |   |
| Description              | The tool to trace execution flow of SlimCORE processor instructions   |
| Features                 | <ul style="list-style-type: none"> <li>▪ tracing the execution of SlimCORE processor instructions (single stepping, dump of register contents with proper indication of register changes)</li> <li>▪ logging of a trace of executed instructions</li> </ul>   |
| Impact                   | <ul style="list-style-type: none"> <li>▪ reverse engineering of unknown SlimCORE instructions</li> </ul>  |

|                  |   |
|------------------|---|
|                  | <ul style="list-style-type: none"> <li>▪ location of SLIM Core instruction sequences implementing DecryptKey functionality</li> </ul> |
| Public reference | <ul style="list-style-type: none"> <li>▪ HITB talk #2 (slides 51-53)</li> </ul>   |
| Deliverables     | <ul style="list-style-type: none"> <li>▪ source code</li> </ul>   |

## II. Notes

1. The tools described above have never been disclosed to any 3rd party. Their existence and usefulness have been known since 2012 though (HITB talk #2).
2. The tools were developed as part of SE-2011-01 project. Their operation was suited to the environment of fully compromised (OS root, JVM root and kernel level access privileges) ITI-2849ST / ITI-2850ST set-top-boxes and SE-2011-01 Proof of Concept code in particular. With the exception of SlimCORE disassembler (standalone tool), successful operation and use of a SlimCORE tracer may require customization and/or porting to the target STi7111 environment (target STB).

## III. Legal Disclaimer

Beside the SRP license, the following paragraphs describe the legal disclaimer for the two reverse engineering tools (THE SOFTWARE) offered.

THE SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW NEITHER SECURITY EXPLORATIONS, ITS LICENSORS OR AFFILIATES, NOR THE COPYRIGHT HOLDERS MAKE ANY REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OR THAT THE SOFTWARE WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS, OR OTHER RIGHTS. THERE IS NO WARRANTY BY SECURITY EXPLORATIONS OR BY ANY OTHER PARTY THAT THE SOFTWARE WILL MEET YOUR REQUIREMENTS OR THAT IT WILL BE ERROR-FREE.

YOU ASSUME ALL RESPONSIBILITY AND RISK FOR THE SELECTION AND USE OF THE SOFTWARE TO ACHIEVE INTENDED RESULTS AND FOR THE INSTALLATION, USE, AND RESULTS OBTAINED FROM IT.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL SECURITY EXPLORATIONS, ITS EMPLOYEES OR LICENSORS OR AFFILIATES BE LIABLE FOR ANY LOST PROFITS, REVENUE, SALES, DATA, OR COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, PROPERTY DAMAGE, PERSONAL INJURY, INTERRUPTION OF BUSINESS, LOSS OF BUSINESS INFORMATION, OR FOR ANY SPECIAL, DIRECT, INDIRECT, INCIDENTAL, ECONOMIC, COVER, PUNITIVE, SPECIAL, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND WHETHER ARISING UNDER CONTRACT, TORT, NEGLIGENCE, OR OTHER THEORY OF LIABILITY ARISING OUT OF THE USE OF OR INABILITY TO USE THE INFORMATION CONTAINED IN THIS DOCUMENT, EVEN IF SECURITY EXPLORATIONS OR ITS LICENSORS OR AFFILIATES ARE ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

## IV. SRP pricing

- SRP AO           NOT AVAILABLE
- SRP EP           NOT AVAILABLE

## APPENDIX A

### Table of contents for a technical document included as part of SRP-2018-01 material:

|  |    |
|--|----|
| INTRODUCTION.....                              | 5  |
| SlimCORE PROCESSOR .....                       | 5  |
| Register Set .....                             | 6  |
| Memory Addressing.....                         | 7  |
| Memory spaces.....                             | 7  |
| Reverse engineering approach .....             | 8  |
| Instruction set.....                           | 12 |
| INTENTIONALLY REMOVED .....                    | 12 |
| INTENTIONALLY REMOVED .....                    | 13 |
| INTENTIONALLY REMOVED .....                    | 13 |
| INTENTIONALLY REMOVED .....                    | 14 |
| INTENTIONALLY REMOVED .....                    | 14 |
| INTENTIONALLY REMOVED .....                    | 15 |
| INTENTIONALLY REMOVED .....                    | 16 |
| INTENTIONALLY REMOVED .....                    | 17 |
| INTENTIONALLY REMOVED .....                    | 20 |
| INTENTIONALLY REMOVED .....                    | 22 |
| INTENTIONALLY REMOVED .....                    | 26 |
| INTENTIONALLY REMOVED .....                    | 26 |
| INTENTIONALLY REMOVED .....                    | 27 |
| INTENTIONALLY REMOVED .....                    | 27 |
| INTENTIONALLY REMOVED .....                    | 29 |
| INTENTIONALLY REMOVED .....                    | 29 |
| Further work .....                             | 30 |
| SlimCORE FIRMWARE .....                        | 31 |
| Locating firmware code and data sections ..... | 32 |
| Magic string and NOP instruction .....         | 32 |
| Kernel symbols.....                            | 33 |
| Firmware architecture .....                    | 34 |
| TKD Crypto core .....                          | 35 |
| Commands and configuration variables .....     | 37 |

|  |    |
|--|----|
| Firmware operation .....                             | 39 |
| STK commands' groups .....                           | 42 |
| Core routines related to CWPK and CWs handling ..... | 43 |
| Crypto DMA handling.....                             | 47 |
| Original reverse engineering annotations .....       | 50 |
| Recent firmware changes .....                        | 51 |
| TKD commands obfuscation .....                       | 52 |
| Prolog and epilog routines.....                      | 53 |
| New commands .....                                   | 54 |
| Potential vulnerabilities and further research.....  | 58 |
| INTENTIONALLY REMOVED .....                          | 58 |
| INTENTIONALLY REMOVED .....                          | 59 |
| INTENTIONALLY REMOVED .....                          | 59 |
| INTENTIONALLY REMOVED .....                          | 60 |
| INTENTIONALLY REMOVED .....                          | 61 |
| INTENTIONALLY REMOVED .....                          | 61 |
| INTENTIONALLY REMOVED .....                          | 62 |
| INTENTIONALLY REMOVED .....                          | 62 |
| INTENTIONALLY REMOVED .....                          | 62 |
| INTENTIONALLY REMOVED .....                          | 63 |
| INTENTIONALLY REMOVED .....                          | 65 |
| TOOLS.....   | 66 |
| SlimCORE disassembler.....                           | 66 |
| Description .....                                    | 66 |
| Sample uses .....                                    | 66 |
| SlimCORE tracer .....                                | 68 |
| Tracer API.....                                      | 68 |
| Description .....                                    | 69 |
| Sample uses .....                                    | 74 |
| REFERENCES.....                                      | 75 |
| APPENDIX A.....                                      | 76 |

## Screenshots of sample pages from a technical document included as part of SRP-2018-01 material:

Notation:  
OR reg1,reg2,#imm

Description:  
Perform logical OR of the contents of register reg2 and an immediate operand and store the result in register reg1.

JMP - Jump register

|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 |
| 1  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 0  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x | x | x | x | x | x |

Notation:  
JMP reg

Description:  
Unconditionally jump to target location given by the contents of register operand.

AND - Logical AND

|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 |
| 0  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1 | 1 | 1 | 1 | 1 | 1 |

Notation:  
AND reg1, reg2, #off

Description:  
Perform logical AND of the contents of registers reg1 and reg2 and store the low register result.

MOVZX - Move to register and zero extend

|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 |
| 0  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1 | 1 | 1 | 1 | 1 | 1 |

Notation:  
MOVZX reg1, reg2, #off

Description:

|        |  |
|--------|--|
| 0x4048 | (checked by STK cmds 0x03, 0x10 and 0x11)<br>state flag indicating STK cmd 0x05 was executed (checked by STK cmds 0x01, 0x02, 0x04 and 0x08) |
| 0x404C | TKD operation mode<br>• 0x01 tld is active<br>• 0x02 dma is active   |
| 0x4050 | state flag indicating STK cmd 0x11 (checked by STK cmd 0x12)   |
| 0x4054 | state flag indicating STK cmd 0x2 (checked by STK cmds 0x21, 0x24)   |
| 0x4060 | SW counter   |
| 0x4064 | number of packets for DMA transfer   |
| 0x4120 | bit idx of current stack frame   |
| 0x4124 | bit idx of next stack frame  |

Table 3 SlimCORE firmware configuration / state variables.

Firmware operation  
Generic schema of a SlimCORE firmware operation is illustrated on Fig. 14.




Fig. 14 SlimCORE firmware operation (STTKDMA-REL\_3.1.6).

SlimCore firmware processes STK commands and issues corresponding TKD commands directly to TKD Crypto core. The results of STK commands (if any) are written back to the arguments buffer.

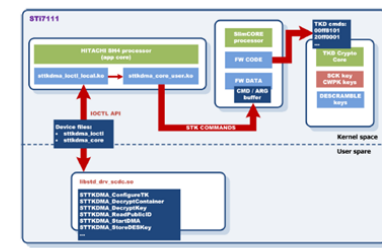


Fig. 10 SlimCORE firmware architecture (associated components and APIs).

STK commands are issued as a response to IOCTL calls received by sttkdma\_ioctl\_local.io device driver from user space library by the means of special device files.

TKD Crypto core  
TKD Crypto core is the main core of STI7111 SoC responsible for all cryptographic and key storage related operations. The core is controlled by the means of 32-bit TKD commands and associated arguments being sent to an I/O port.

TKD Crypto Core supports the following ciphers:

- TDES\_ECB\_128
- AES\_ECB\_128
- AES\_CBC\_128
- AES\_CTR\_128

Generic format of a TKD command is presented on Fig. 11.

[http://sttkdma\\_ioctl1/](http://sttkdma_ioctl1/) / [http://sttkdma\\_core\\_for\\_ITI-38495T\\_and\\_ITI-29500T\\_set-top-boxes/](http://sttkdma_core_for_ITI-38495T_and_ITI-29500T_set-top-boxes/)

## Document features

- 70+ pages of unpublished and in-depth technical information
- information regarding SlimCORE processor instruction set, their opcodes format and processor operation
- information about reverse engineering approach taken to discover unknown SlimCORE instructions' opcodes and their meaning
- information regarding SlimCORE firmware architecture and operation (threads and their dispatching, STK / TKD commands and their groups, crypto DMA)
- information about locating SlimCORE firmware code and data sections in STTKDMA device driver file and kernel memory
- broad scope of firmwares covered (STTKDMA-REL 3.1.6, 3.5.0 and 3.9.2) making it easy to analyze any other SlimCORE firmware for STI7111 SoC
- detailed explanation of selected code parts (i.e. CWPK and CW keys handling, initialization, crypto DMA)
- information regarding firmware configuration and state variables
- information about changes to most recent firmware 3.9.2 and STMicroelectronics' "addressing" of security issues from 2012
- information about unverified, potential vulnerabilities and further research directions pertaining to TKD crypto core and STI7111 security
- description of SlimCORE disassembler and tracer tools

**List of figures from a technical document included as part of SRP-2018-01 material:**

- Fig. 1 SlimCORE location in STi7111 SoC.
- Fig. 2 SlimCORE registers.
- Fig. 3 Output buffer of a GetPublicID command.
- Fig. 4 SlimCORE instruction sequence corresponding to GetPublicID result.
- Fig. 5 Running user provided code as part of GetPublicID code path.
- Fig. 6 MOV instructions patterns.
- Fig. 7 CMP and conditional jump instructions patterns.
- Fig. 8 SlimCORE firmware loading code.
- Fig. 9 SlimCORE firmware offsets in chipset memory space.
- Fig. 10 Firmware code location and a magic string.
- Fig. 11 SlimCORE firmware architecture (associated components and APIs).
- Fig. 12 Generic TKD command format.
- Fig. 13 Lower 16 bits of a TKD command for standard DMA operation.
- Fig. 14 Lower 16 bits of a TKD command for SCK DMA operation.
- Fig. 15 SlimCORE firmware operation (STTKDMA-REL\_3.1.6).
- Fig. 16 A response to Conax CAS EMM message carrying chipset pairing information.
- Fig. 17 Deobfuscation of TKD commands.
- Fig. 18 Internal SlimCORE processor structure.
- Fig. 19 SlimCORE tracer architecture.
- Fig. 20 Tracer's core routine implementation.

**List of tables from a technical document included as part of SRP-2018-01 material:**

- Table 1 Instructions for data exchange with Crypto TKD core.
- Table 2 The mapping of STK commands to TKD commands.
- Table 3 SlimCORE firmware configuration / state variables.
- Table 4 STK commands groups and their description.
- Table 5 TKD DMA configuration commands.
- Table 6 SlimCORE firmware versions and their differences.
- Table 7 Summary of operations implemented by STK command 0x44.
- Table 8 Customer mode values and corresponding STK commands.
- Table 9 Customer mode value and special handling of STK commands.
- Table 10 Command line arguments of SCDisasm tool.
- Table 11 Tracer's API subroutines.
- Table 12 Tracer's API variables.
- Table 13 Tracer's state variables.
- Table 14 Translation rules used by the instruction rewriter.

## APPENDIX B

### A fragment of reverse engineering annotations for STTKDMA-REL\_3.1.6:

```

!/*## (c) SECURITY EXPLORATIONS    2011 poland                #*/
!/*##    http://www.security-explorations.com                #*/
!
!/* RESEARCH MATERIAL:    SRP-2018-01
*/
!/* Reverse engineering annotations                            */
!/* SlimCORE firmware ver    : STTKDMA-REL_3.1.6            */
!/*    code size: 5852 (0x16dc)                                */
!/*    sha-1    : afe518789d1b0b1d3c0f8efd2704ac84a69140ed */
!
!/* THIS SOFTWARE IS PROTECTED BY DOMESTIC AND INTERNATIONAL COPYRIGHT LAWS */
!/* UNAUTHORISED COPYING OF THIS SOFTWARE IN EITHER SOURCE OR BINARY FORM IS */
!/* EXPRESSLY FORBIDDEN. ANY USE, INCLUDING THE REPRODUCTION, MODIFICATION, */
!/* DISTRIBUTION, TRANSMISSION, RE-PUBLICATION, STORAGE OR DISPLAY OF ANY */
!/* PART OF THE SOFTWARE, FOR COMMERCIAL OR ANY OTHER PURPOSES REQUIRES A */
!/* VALID LICENSE FROM THE COPYRIGHT HOLDER.                    */
!
!/* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS */
!/* OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, */
!/* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL */
!/* SECURITY EXPLORATIONS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, */
!/* WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF */
!/* OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE */
!/* SOFTWARE.                                                  */
0 #####
0 DISPATCH idx 0x04 -> 0x2000000 (init code)
0 #####
b ;counter = 0
c ;memory idx of 0x4040 addr
f ;store 0 to [0x4040-0x4060]
10 ;chip customer mode
12 ;low nibble of chip customer mode
14 ;-> chip customer mode == 0x05
16 ;-> chip customer mode == 0x02
18 ;-> chip customer mode == 0x06 (OUR CASE)
1a ;-> chip customer mode == 0x0b
1c ;-> chip customer mode == 0x0f
1e ;-> chip customer mode == 0x03
20 ;-> chip customer mode == 0x07
22 ;-> chip customer mode == 0x08
24 ;-> chip customer mode == 0x0c
26 ;05 -> 0x02 as customer mode
28 ;02 -> 0x04 as customer mode
2a ;06 -> 0x05 as customer mode
2c ;0b -> 0x08 as customer mode
2e ;0f -> 0x09 as customer mode
30 ;03 -> 0x10 as customer mode
32 ;07 -> 0x11 as customer mode
34 ;08 -> 0x20 as customer mode
36 ;0c -> 0x21 as customer mode
38 ;store customer mode =(0x02,0x04,0x05,0x08,0x09,0x10,0x11,0x20,0x21,*0x40)
39 ;subroutine return addr
3a ;init all of the keys (CWPK, CWs)
...

```