hgi

“Everything That Has A Beginning Has An End”

The Matrix Revolutions

.

bl4ckic3 ali@ali.re

ScepticCtf tobias.scharnowski@rub.de

RUHR UNIVERSITÄT BOCHUM RUB

Special Thanks

- Thorsten Holz, Ruhr-University Bochum
- Thomas Weber, Sec-Consult
- Alexandre Gazet, Airbus Cyber Security
- Marina Krotofil, BASF
- Lucian Cojocar from VU Amsterdam
- Nikita Golovliov, TU Eindhoven

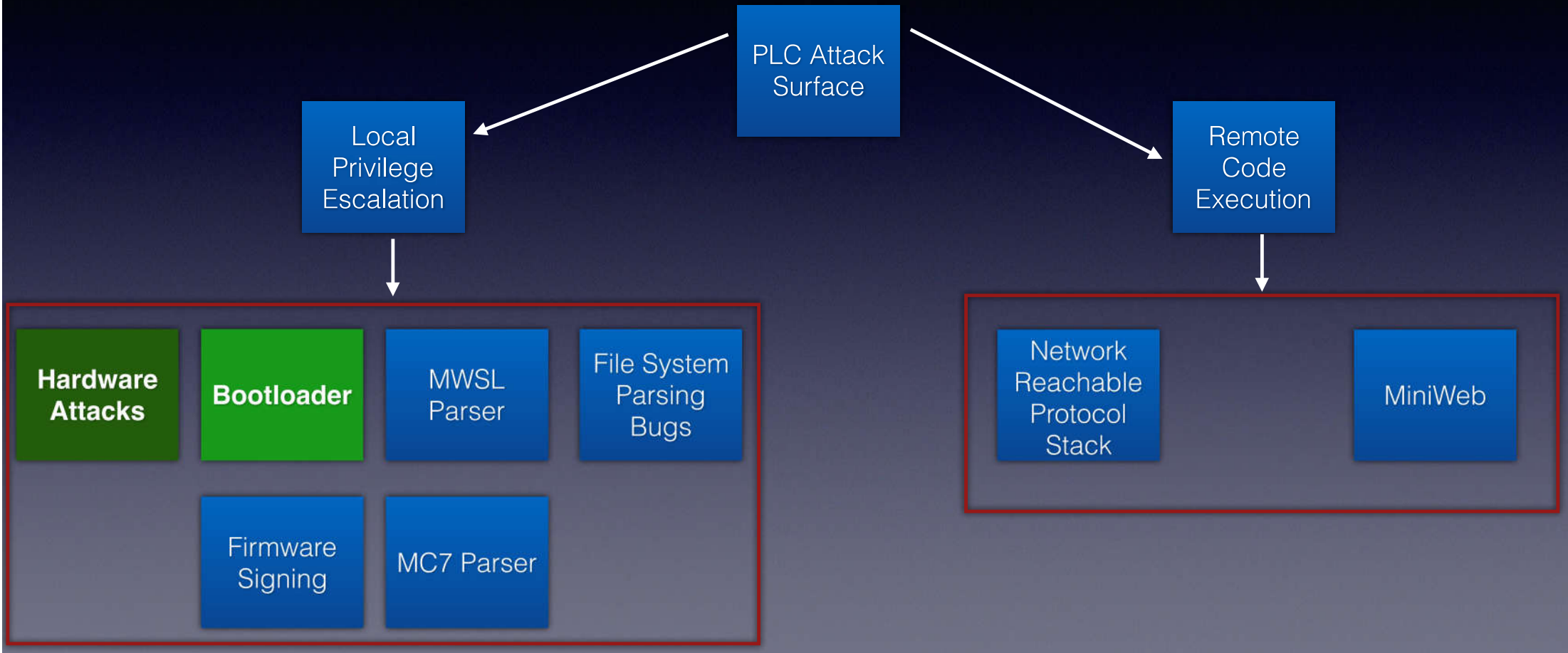
Conclusions

- PLCs are becoming more complex
- Vendors are introducing security measures to their devices
- Vendors have legacy access features that undermine their product security
- Researchers/Vendors disclosure dynamics
- Vendors should rethink the security via obscurity mindset
- There are a lot of things to be done to make PLCs safer

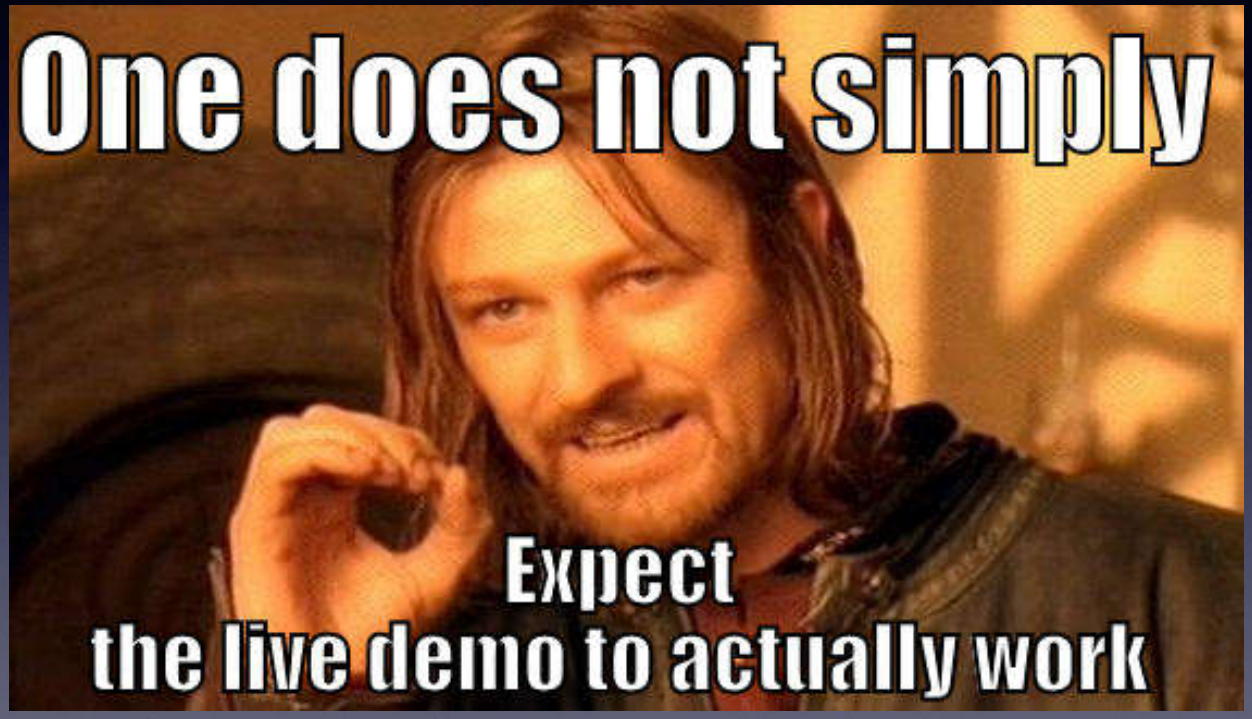
Client utility is available

- A utility code to interact with bootloader using special access feature
- Currently supporting bootloader v4.2.1 of the Siemens S7-1200v4.

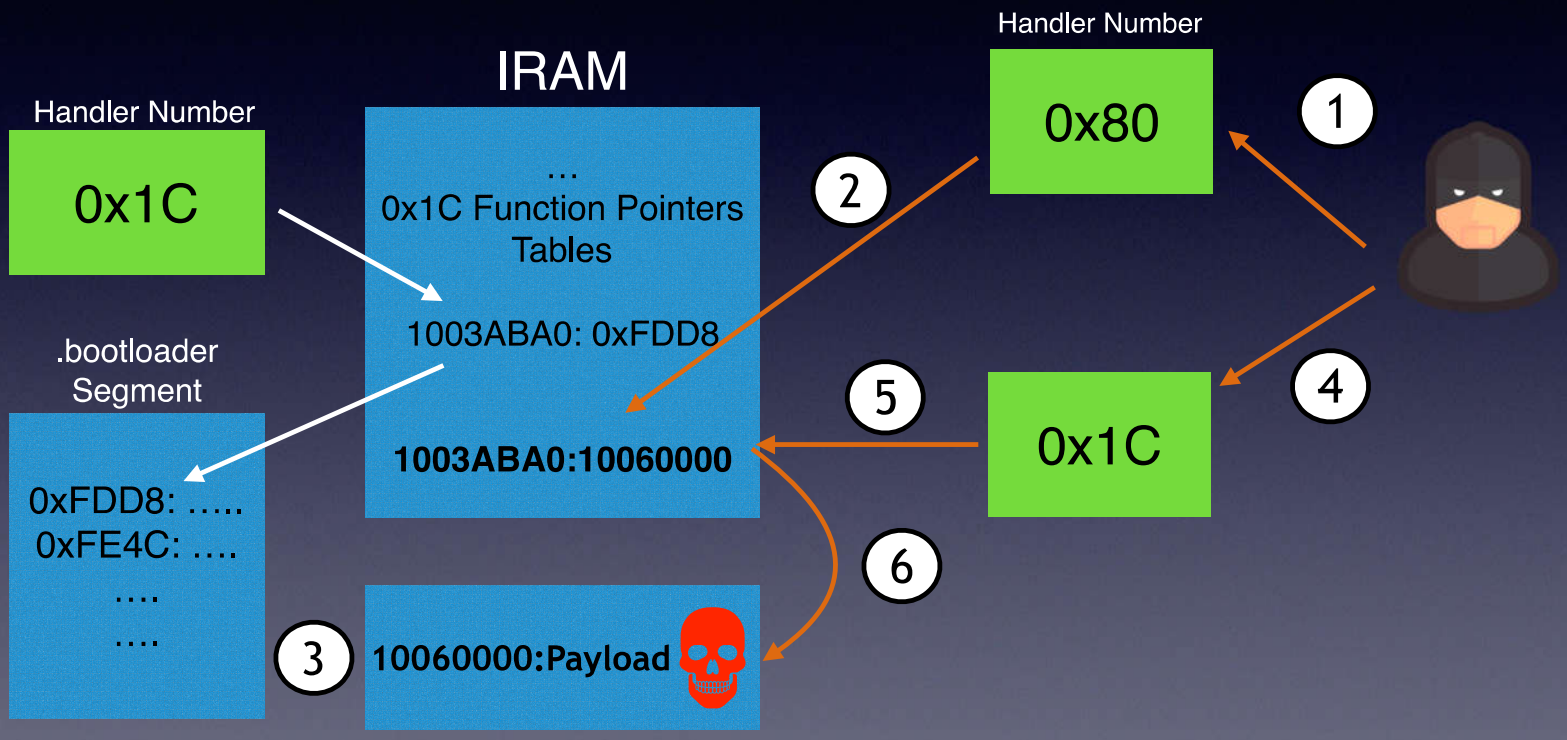
What else is out there?



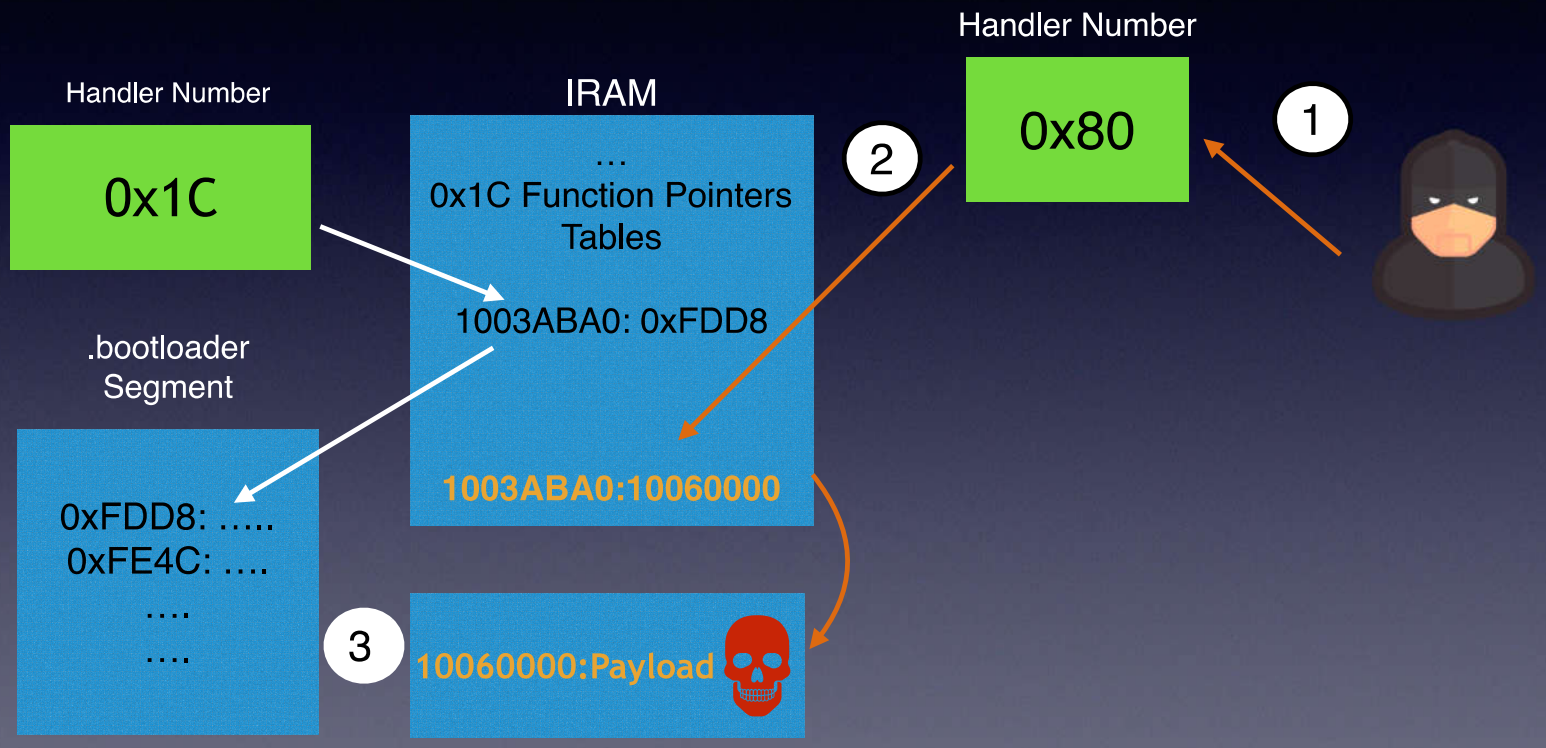
DEMO



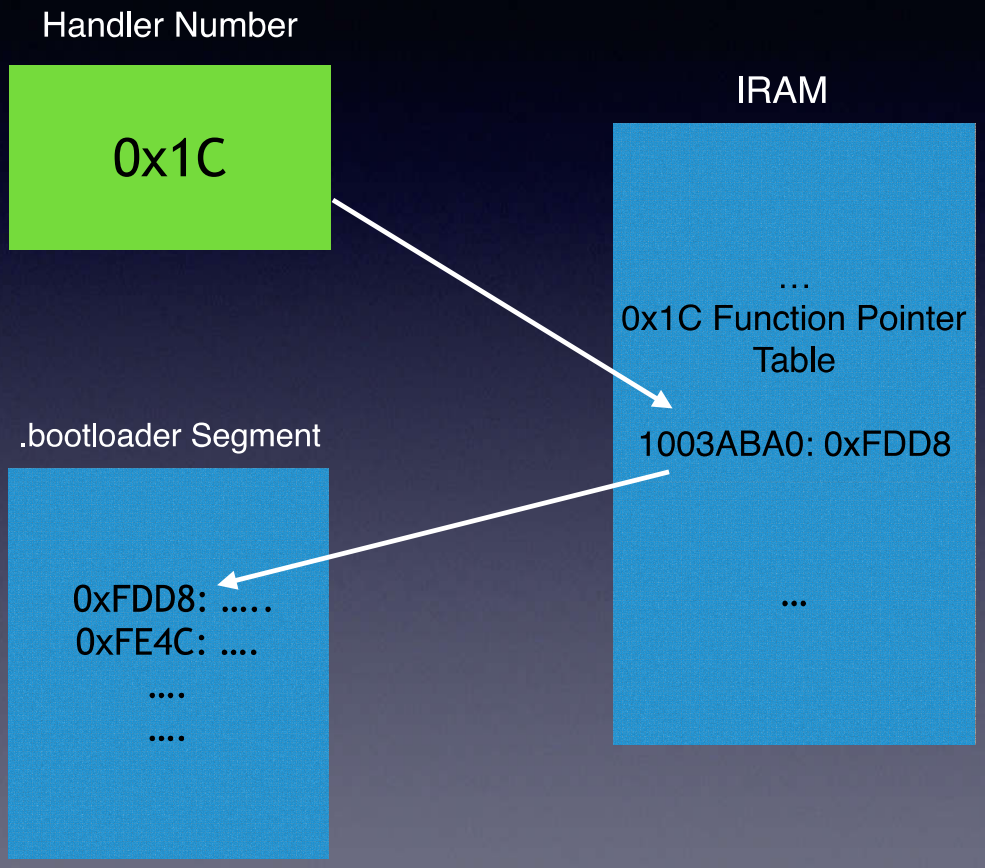
Siemens S7-1200/S7-200 SMART Bootloader Arbitrary Code Execution



Siemens S7-1200/S7-200 SMART Bootloader Arbitrary Code Execution



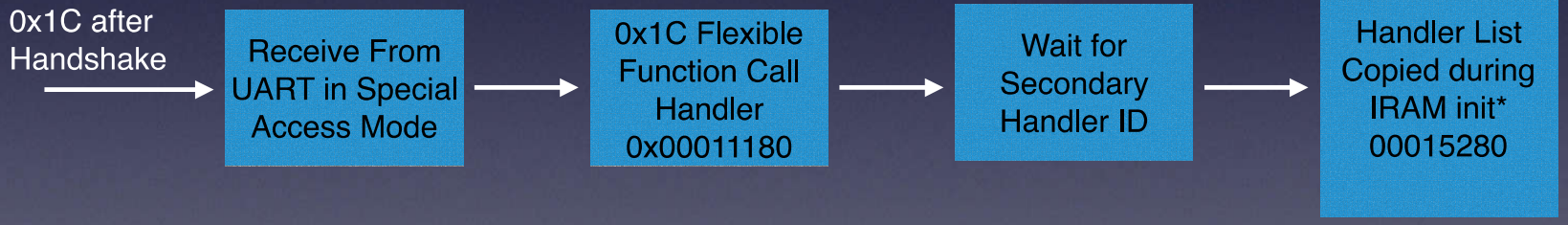
Siemens S7-1200/S7-200 SMART Bootloader Arbitrary Code Execution



What can we do with it?

0x1C Primary Handler

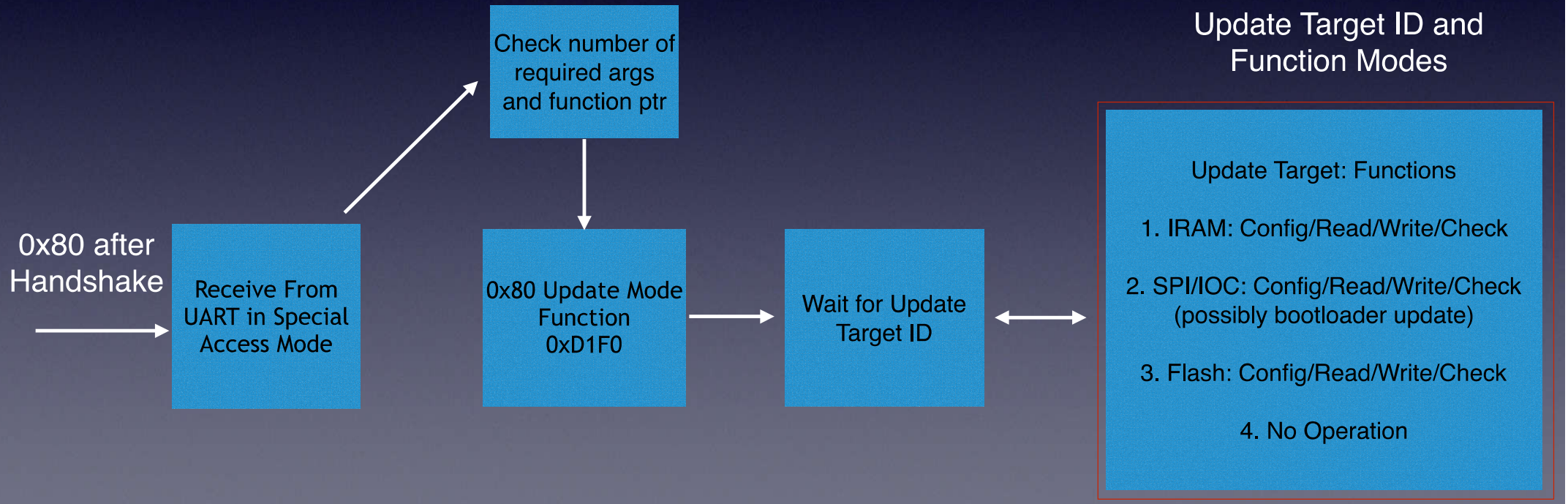
- This handler allows to call functions from a secondary list.




```
                ; bootloader:off_6E0fo
bl_UART_add_hook_00_some_setup
0
bl_UART_add_hook_01_perform_flash_write_list
0
bl_UART_add_hook_02_some_fill_arg_list
0
bl_UART_add_hook_03_set_80568300
0
nullsub_152
0
bl_UART_add_hook_05_get_80280000_byte_2
0
bl_UART_add_hook_06_query_some_dw
0
bl_UART_add_hook_07_query_some_dw
0
bl_UART_add_hook_08_wipe_chosen_flash_offset
0
bl_UART_add_hook_09_read_some_chosen_second_byte
0
bl_UART_add_hook_0a_write_crc_or_metadata_word
0
bl_UART_add_hook_0b_count_actual_data_dwords
7
bl_UART_add_hook_0c_count_some_configure_or_erase
0
bl_UART_add_hook_0d_some_find_dword_occurrences
0
bl_UART_add_hook_0e_some_read_flash
0
nullsub_153
0
bl_UART_add_hook_10_write_flash_with_incr_values
0
bl_UART_add_hook_11_read_fw_flash
3
bl_UART_add_hook_12_call_count_flash_occurrences
3
bl_UART_add_hook_13_some_wipe_flash
3
bl_UART_add_hook_14_call_count_flash_occurrences
3
bl_UART_add_hook_15_some_do_flash_update
3
bl_UART_add_hook_16_some_do_multiple_flash_updates
3
bl_UART_add_hook_17_call_count_flash_occurrences
3
bl_UART_add_hook_18_multi_flash_reads
0
bl_UART_add_hook_19_nullsub_154
0
bl_UART_add_hook_1a_nullsub_155
```

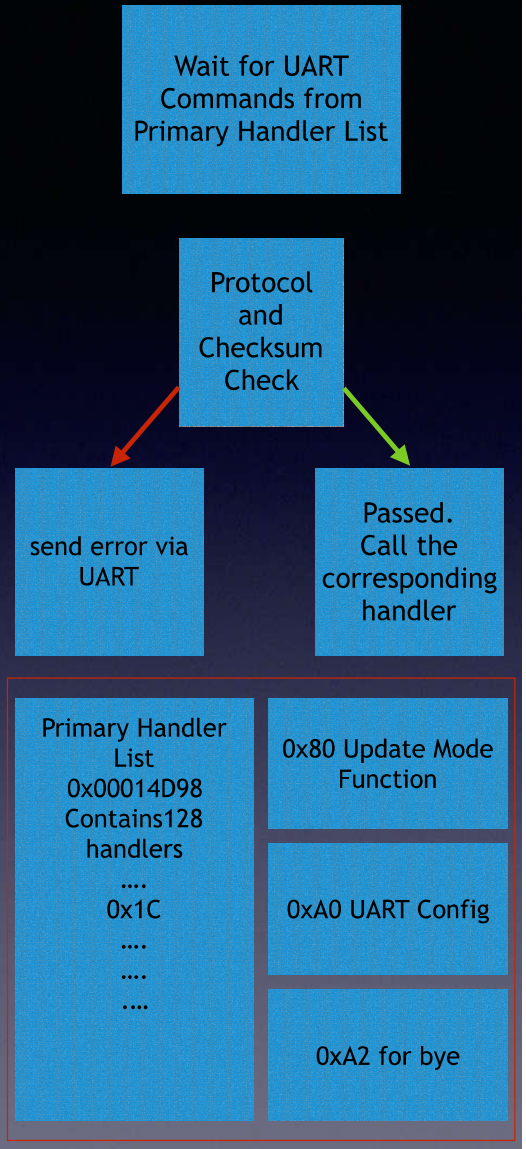
0x80 Handler, Update Mode Function

- 0x80 allows us to write to IRAM in update target 1 (IRAM)



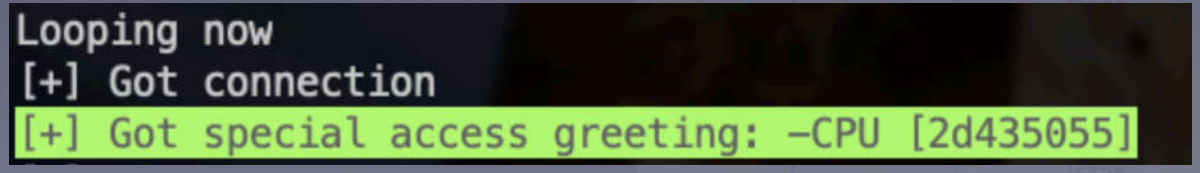
Primary Handlers After Handshake

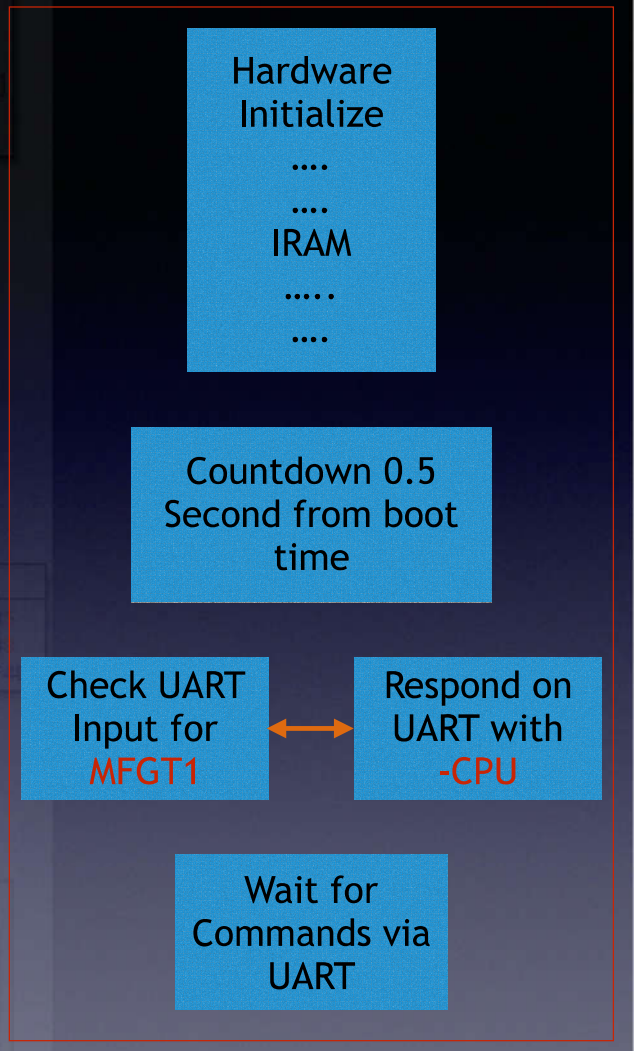
Index	Input Length	Functionality
0	2	Get bootloader Version
1	2	Get firmware version
2	2	Check bootloader CRC
3	2	Check CRC flash part 2 (sectors 0 to 0x203), returns details on failure
4	2	Check IRAM (internal RAM) read/write
5	xx	Check TCM (Tightly Coupled Memory) read/write
6	3	Hardware tests for I2C, IOC, MAC
7	2	Some low level reset involving a temporay fiq handler
8	2	Some functionality related to I2C2
0c	6	Perform some read and crc calculation on flash memory, uses buf also used with ADC IOC
0d	2	Performs reads of static bootloader values and writes to some hardware mapped addresses
0e	3	Outputs some internal RAM values previously retrieved from static bootloader contents
11	2	Something related to MMC looping/checking
12	2	Some activation or waiting for I2C0
14	7	[!] Print (and seemingly update) current flash contents to(/from) UART.
16	2	Queries and returns some MMC bit.
17	2	Performs some lookup table based stuff in I2C1
18	3	Prepare reading flash update via UART to IRAM
19	>6	Read flash update via UART to IRAM
1a	2	[!] Commit flash update
1c	XX	**We will talk about it later**
30	XX	Query some CRC info about flash mode/part 2
31	7	[!] Performs an update of 8 bytes of flash number 2. This may be sending a length/crc pair
32	2	Check flash 2
..	xx	Prepares writes to EMB0 (plus some more ops on state structs).



Special Access Feature

- Bootloader initializes the hardware
  - This includes copying some content from bootloader to IRAM.
- Only wait half a second from UART for :
  - **MFGT1** strings, possibly **M**it **F**reundlichen **G**rußen (German Greeting).
  - If PLC received “MFGT1” string it will acknowledge with **-CPU**
  - PLC now waits for Special Access Feature commands at 0xedf8.



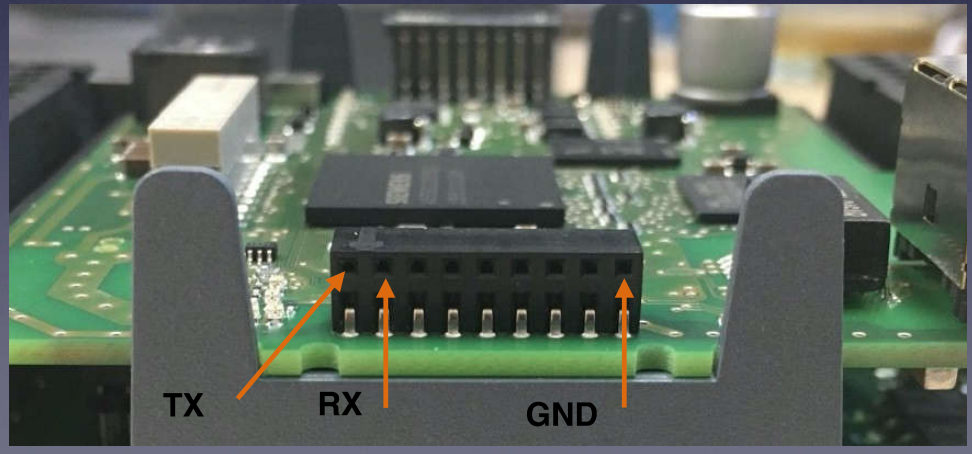


UART Packet Format:
`<length_byte><contents><checksum_byte>`

Special Access In S7-1200 (inc SiPLUS) & S7-200 SMART

- A set of special functionalities within the PLC bootloader
  - Various diagnostics functions, Exposed via UART Access
  - UART port previously documented*.
- Addresses in this presentation are from bootloader v4.2.1

```
[+] Successfully turned on power supply
Looping now
[+] Got connection
[+] Got special access greeting: -CPU [2d435055]
[*] sending packet: 0200fe
[+] Got Siemens bootloader version: V4.2.3
[*] sending packet: 04803bc27f
Writing 0000/0124
[*] sending packet: 18845a2e00030100ffffffffffffff
```



*Cojocar et al. Off-the-shelf Embedded Devices as Platforms for Security Research

Special Access Feature in Siemens S7 PLCs

Undocumented HTTP Handlers on ADONIS MiniWeb

HTTP GET Request Handler	Description
/appapiappa/vvvvvvvvv	version
/appapiappa/vvvvvvvvvv	version
/appapiappa/lililili	log in
/appapiappa/lolololololo	log out
/appapiappa/cmcmcmcmcm	change CPU mode
/appapiappa/flflflflfl	flash LEDs
/appapiappa/gbpigbpigb	get station info as json (name, mac, mode)
/appapiappa/gmigmigmig	get module info (list): get name, serial, FW version, HW version, status
/appapiappa/galegalega	unknown
/appapiappa/galeldgale	get AS log entry
/appapiappa/gvigvigvig	unknown
/appapiappa/svsvsvsvsv	unknown

Firmware Update Process On S7 PLC

- Updates available via Webserver and SDCard (24MB, costs a whopping ~250 Euros!)
- Siemens S7-1200 firmware (.upd) are compressed update structure, contains:
  - Constant size metadata (44 bytes)
  - Headers describing contents of the file
  - Contents of headers
    - Types seen so far:BG_ABL: descriptor, FW_SIG: firmware signature, A00000: Main update contents, B00000: metadata

ADONIS TCP/IP Stack

udpsock.c	16 KB	C Source
tcpsack.c	11 KB	C Source
tcpport.c	2 KB	C Source
tcpip.h	4 KB	C Head...Source
tcp_zio.c	7 KB	C Source
tcp_var.h	14 KB	C Head...Source
tcp_usr.c	17 KB	C Source
tcp_timr.h	6 KB	C Head...Source
tcp_timr.c	13 KB	C Source
tcp_subr.c	16 KB	C Source
tcp_seq.h	3 KB	C Head...Source
tcp_out.c	34 KB	C Source
tcp_menu.c	7 KB	C Source
tcp_in.c	67 KB	C Source
tcp_fsm.h	5 KB	C Head...Source
soselect.c	10 KB	C Source
socket2.c	23 KB	C Source
socket.c	35 KB	C Source
sockcall.c	26 KB	C Source
rawsock.c	13 KB	C Source
protosw.h	10 KB	C Head...Source
nptcp.c	43 KB	C Source

- ADONIS utilizes the InterNiche Technologies TCP/IP Stack v3.1
- IP, TCP, UDP, ARP, SNMP, NTP are supported.
- TCP/IP stack is already leaked via some OEM of the stack. Not all components are used in S7 PLCs.
- We were able to independently compile the leaked stack.

```
00004A  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\tcp_usr.c
00004B  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\tcp_subr.c
00004A  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\tcp_out.c
000049  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\tcp_in.c
00004B  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\soselect.c
00004A  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\socket2.c
000049  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\socket.c
00004B  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\sockcall.c
000048  C  C:\\\\Sources\\\\fwf_update1\\\\s7p.comm\\\\TCIP\\\\src_iniche_core\\\\tcp\\\\nptcp.c
```

Siemens Firmware Boot Process

- **0x00040040**: Enable instruction cache and some system specific coprocessor register
- **0x000400B4**: Set up stack pointers for different execution modes.
- **0x000402A4**: Set up the Vectored Interrupt Controller (VIC)
- **0x000400B4**: Configure MPU via MPU coprocessor instructions.
- **0x000403E4**: Zero out the .bss section
- **0x00567698**: Set some IOC config and run ADONIS boot function

All addresses in this presentation are for Firmware v4.02.01 on a 6ES7212-1AE40-0XB0 PLC

I/O Memory Mapping on A5E30235063, From 0x00df4a80

Segment Name	Range	Size	Segment Name	Range	Size
itcm	0x00000000-0x00008000	0x00008000	MAP3_OUTPUTS	0xfffba000-0xfffba218	0x00000218
ddram	0x00008000-0x04000000	0x03ff8000	MAP3_ITIMER0	0xfffbb000-0xfffbb010	0x00000010
configured_dtcm	0x10010000-0x10014000	0x00004000	MAP3_ITIMER1	0xfffbb010-0xfffbb020	0x00000010
internal_ram0	0x10030000-0x10040000	0x00010000	MAP3_ITIMER2	0xfffbb020-0xfffbb030	0x00000010
internal_ram1	0x10040000-0x10050000	0x00010000	MAP3_ITIMER3	0xfffbb030-0xfffbb040	0x00000010
MAP3_PWRSTK	0xfffb0000-0xfffb003c	0x0000003c	MAP3_ITIMER4	0xfffbb040-0xfffbb050	0x00000010
MAP3_SPI0	0xfffb1000-0xfffb1018	0x00000018	MAP3_ITIMER5	0xfffbb050-0xfffbb060	0x00000010
MAP3_SPI1	0xfffb2000-0xfffb2018	0x00000018	MAP3_ITIMER6	0xfffbb060-0xfffbb070	0x00000010
MAP3_I2C0	0xfffb3000-0xfffb306c	0x0000006c	MAP3_ITIMER7	0xfffbb070-0xfffbb080	0x00000010
MAP3_I2C1	0xfffb4000-0xfffb406c	0x0000006c	MAP3_ITIMER8	0xfffbb080-0xfffbb090	0x00000010
MAP3_I2C2	0xfffb5000-0xfffb506c	0x0000006c	MAP3_ITIMER9	0xfffbb090-0xfffbb0a0	0x00000010
MAP3_ADC	0xfffb6000-0xfffb6024	0x00000024	MAP3_ITIMER10	0xfffbb0a0-0xfffbb0b0	0x00000010
MAP3_UART0	0xfffb7000-0xfffb709c	0x0000009c	MAP3_ITIMER11	0xfffbb0b0-0xfffbb0c0	0x00000010
MAP3_UART1	0xfffb8000-0xfffb809c	0x0000009c	MAP3_ITIMER12	0xfffbb0c0-0xfffbb0d0	0x00000010
MAP3_HSC0	0xfffb9100-0xfffb9180	0x00000080	MAP3_ITIMER13	0xfffbb0d0-0xfffbb0e0	0x00000010
MAP3_HSC1	0xfffb9180-0xfffb9200	0x00000080	MAP3_TIMERS	0xfffbb000-0xfffbb15c	0x0000015c
MAP3_HSC2	0xfffb9200-0xfffb9280	0x00000080	MAP3_IOC	0xfffbc000-0xfffbc02c	0x0000002c
MAP3_HSC3	0xfffb9280-0xfffb9300	0x00000080	MAP3_FL_MEMCTL	0xfffbd000-0xfffbe000	0x00001000
MAP3_HSC4	0xfffb9300-0xfffb9380	0x00000080	MAP3_VIC	0xfffffc00-0xfffffe00	0x00000200
MAP3_HSC5	0xfffb9380-0xfffb9400	0x00000080	MAP3_EMB0	0xfff50000-0xfff50048	0x00000048
MAP3_INPUTS	0xfffb9000-0xfffb9400	0x00000400	MAP3_EMB1	0xfff51000-0xfff51048	0x00000048
MAP3_PLS0	0xfffba000-0xfffba080	0x00000080	MAP3_DDR_MEMCTL	0xfff52000-0xfff5208c	0x0000008c
MAP3_PLS1	0xfffba080-0xfffba100	0x00000080	MAP3_MMC	0xfff60000-0xfff60104	0x00000104
MAP3_PLS2	0xfffba100-0xfffba180	0x00000080	MAP3_LCD	0xfff70000-0xfff70ff8	0x00000ff8
MAP3_PLS3	0xfffba180-0xfffba200	0x00000080	MAP3_MAC	0xfff90000-0xfff900a4	0x000000a4
NOT USED	NOT USED	NOT USED	MAP3_BOOL_HELPER	0xfffa0000-0xfffa4000	0x00004000

All addresses in this presentation are for Firmware v4.02.01 on a 6ES7212-1AE40-0XB0 PLC

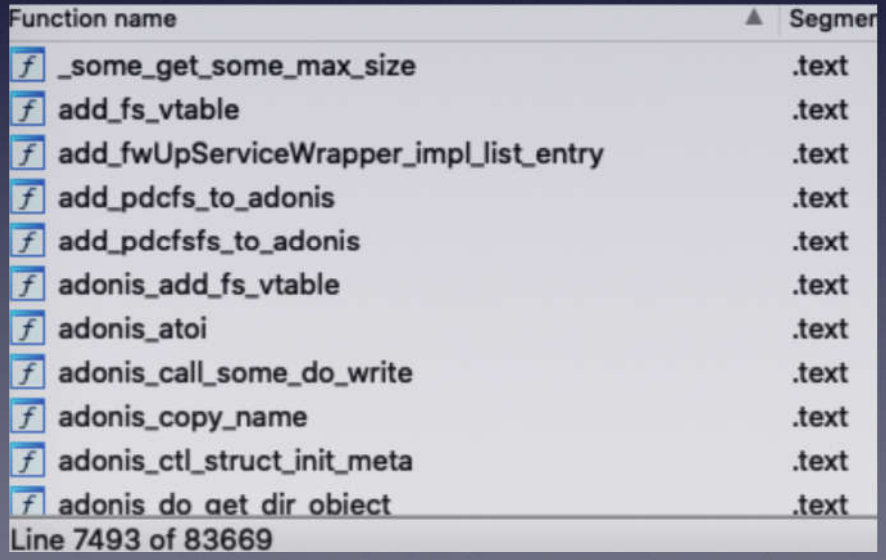
Firmware Memory Mapping on S7-1200 v4, from 0x000439C0

Segment Name	Range	Size	Flags
.exec_in_lomem	0x00000000-0x000075b4	000075b4	1
.bitable	0x00040000-0x00040040	00000040	1
.sdramexec	0x00040040-0x00040510	000004d0	1
.syscall	0x00040540-0x00040548	00000008	1
.th_initial	0x00041040-0x00043998	00002958	33
.secinfo	0x000439c0-0x00043cfc	0000033c	34
.fixaddr	0x00043d00-0x00043d00	00000000	4
.fixtype	0x00043d00-0x00043d00	00000000	4
.text	0x00043d00-0x00defda0	00dac0a0	33
.rodata	0x00defdc0-0x011a116c	003b13ac	34
.data	0x011a1180-0x011c1d14	00020b94	42
.bss	0x01e01040-0x02620f58	0081ff18	12
.cc_memory	0x03641040-0x03641040	00000000	4
.uninitialized	0x03c41040-0x06fac934	0336b8f4	12
CLSI_CACHED_MEM_POOL	0x06fac940-0x06fac940	00000000	4
.dram_uncache	0x07ff0000-0x07ff0000	00000000	4
MAP_MAC_MEM	0x07ff0000-0x07ff0494	00000494	12
.iram0	0x10030000-0x10037aa0	00007aa0	12
.iram1	0x10040000-0x1004c35c	0000c35c	12
.crctable	0x1004f400-0x1004f800	00000400	12
.softboot	0x1004f800-0x1004ff00	00000700	12
.bootinfo	0x1004ff00-0x1004ff1c	0000001c	12
.dtcm	0x10010000-0x10012c70	00002c70	12

All addresses in this presentation are for Firmware v4.02.01 on a 6ES7212-1AE40-0XB0 PLC

Siemens Firmware Dump

- We got a dump of the RAM for both .text and dynamic memory area.
- LZP3 compressed when downloaded from Siemens website
- ~13MB binary.
- ~84000 functions identified.



Siemens AG ADONIS RTOS Components

SIEMENS AG
ADONIS RTOS
0x00040040 at .sdramexec

ADONIS Init Kernel
0x00E7AC50

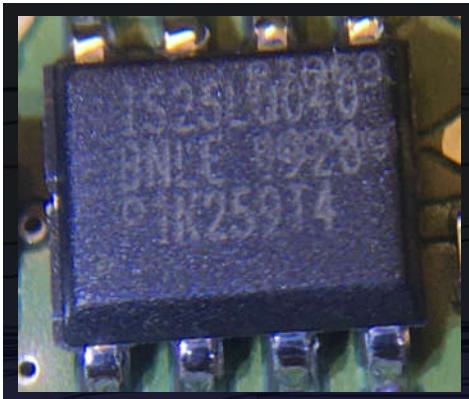
ADONIS Library OS Services

- S7P File Services
- Dinkumware C/C++ Lib v5.01
- PDC FS
- EFS/NTFS
- NicheStack IPV4/ SNMP
- Some Low-Level Configuration (e.g., MPU Reconfiguration)
- MiniWeb Server & MWSL Parser, OpenSSL
- Log/Diag System
- ACE 6.3 for S7P Components ADAPTIVE COMMUNICATION ENVIRONMENT

Automation-related Services

- PNIO PROFINET
- AWP Automated Web Programming
- MC7P Parser
- MC7P RunTime
- MC7P Compiler
- Siemens OMS
- Central IO Subsystem
- ALARM Subsystem
- OMS SP Parser
- OMS Configuration
- OMS FAT

All addresses in this presentation are for Firmware v4.02.01 on a 6ES7212-1AE40-0XB0 PLC



Start

Initialize Hardware Stage 1

Init Bootloader in RAM 0x210

Check bootloader CRC checksum

Initialize Other Hardware Stage 2 SPI, IOC, UART

Wait for a UART Command

Special Access Feature

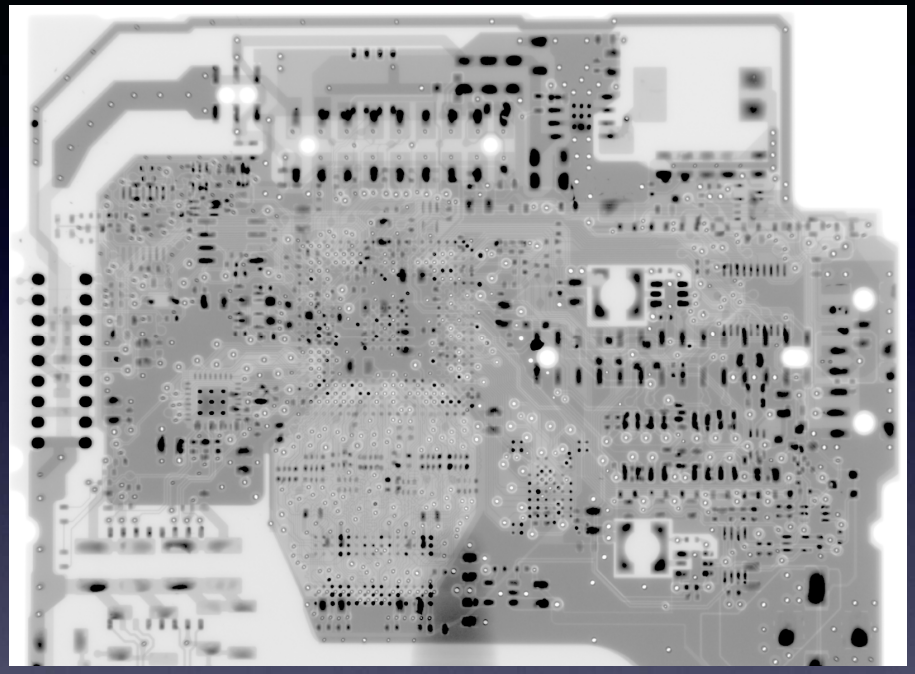
Initialize CRC Table Constants & Page Integrity List

Check CRC of Flash

Final Mapping of Flash Memory to RAM 0x014B74

All addresses for bootloader v4.2.1

3D X-Ray Tomography



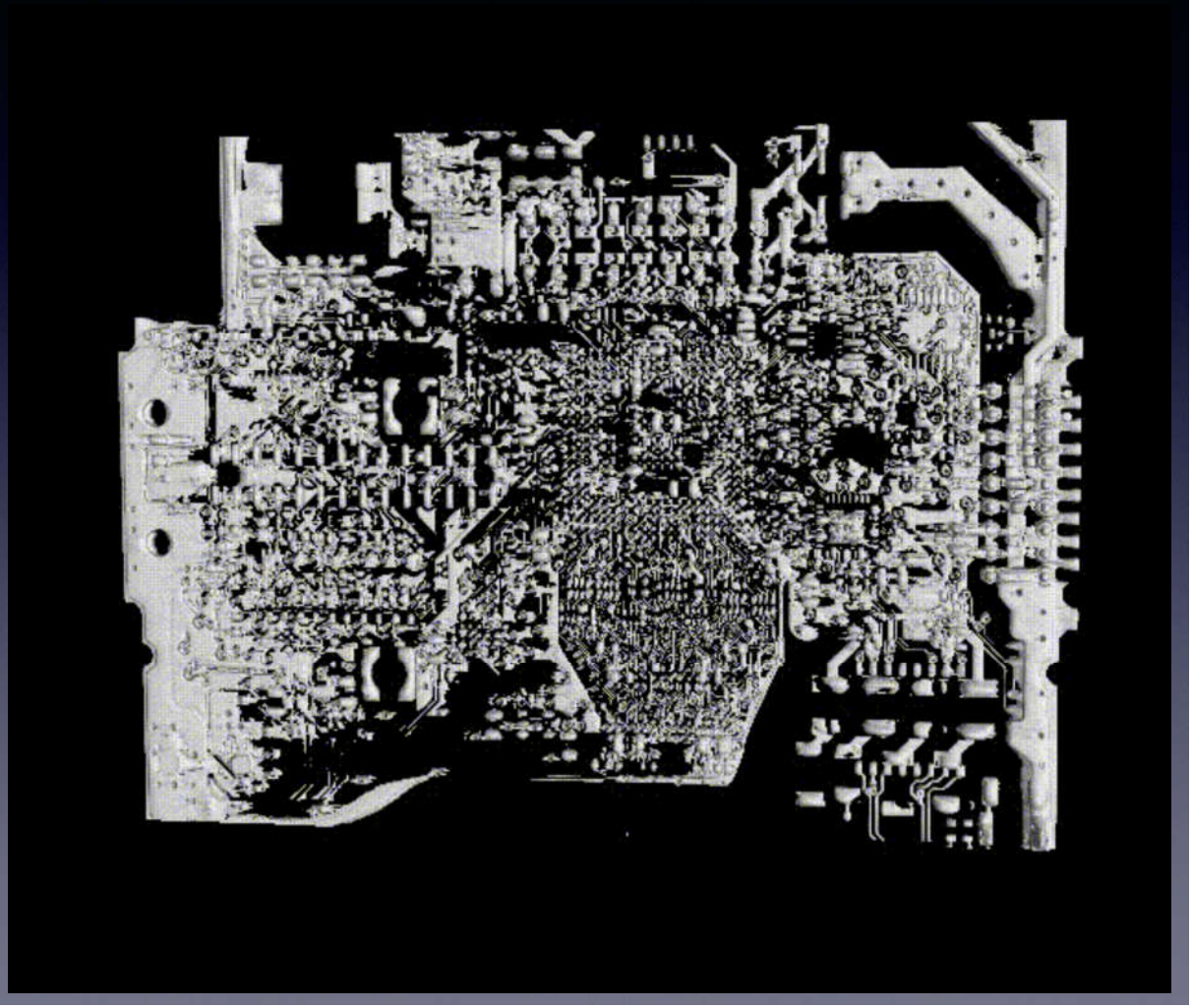
PCB

PCB Connection

VCC

GND

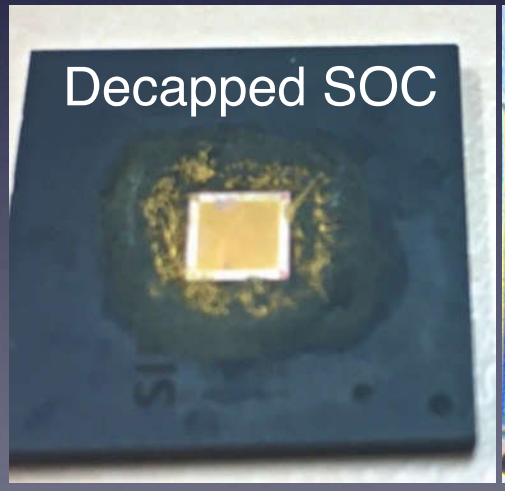
PCB Connection

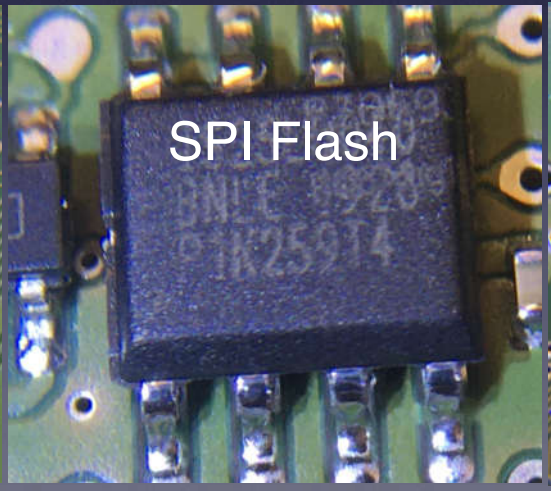


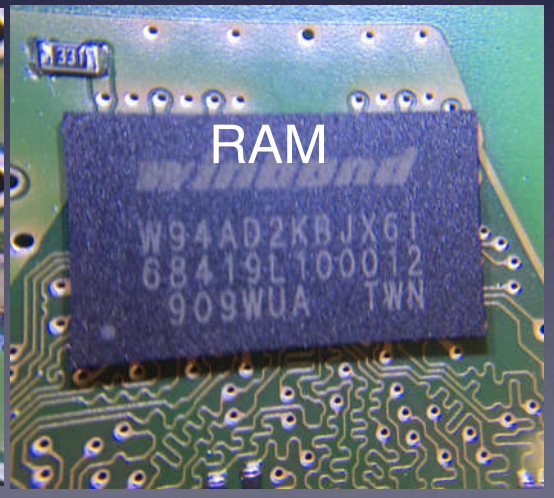
S7-1200 Specs, 3D X-Ray Tomography

S7-1200 v4 Closer Look

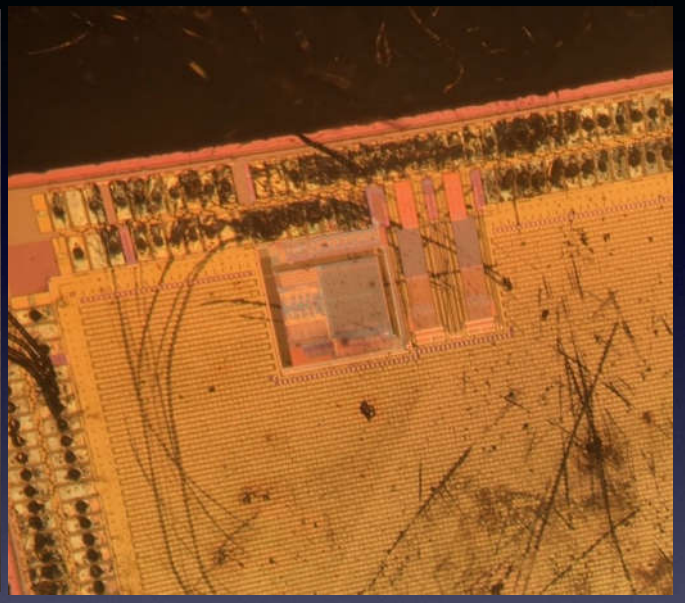
- Cortex R4 Revision 3 ARMv7 R, Big Endian, based on a **Memory Protection Unit (MPU)**
- Multiple RAM Sizes from Wingbond or Micron Technologies
- SPI Flash (multiple types)
- NAND (multiple types)







S7-1200 v4 PLC hardware - SoC Decap



- **Renesas 811005 Manufactured, Siemens A5E30235063 ARM Cortex-R4 (Big-Endian), 2010.**
- **Instruction Set/Read Main ID Register: Running CP15 instruction inside PLC yields 0x411fc143 response.**

To access the Main ID Register, read CP15 with:

```
MRC p15, 0, <Rd>, c0, c0, 0 ; Read Main ID Register
```

Cortex-R4 and Cortex-R4F Technical Reference Manual Revision: r1p3

Home > System Control Coprocessor > System control coprocessor registers > c0, Main ID Register

4.2.2. c0, Main ID Register

The Main ID Register returns the device ID code that contains information about the processor.

The Main ID Register is:

- a read-only register
- accessible in Privileged mode only.

Source:http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0360f/I65012.html

S7-1200 V4 PLC

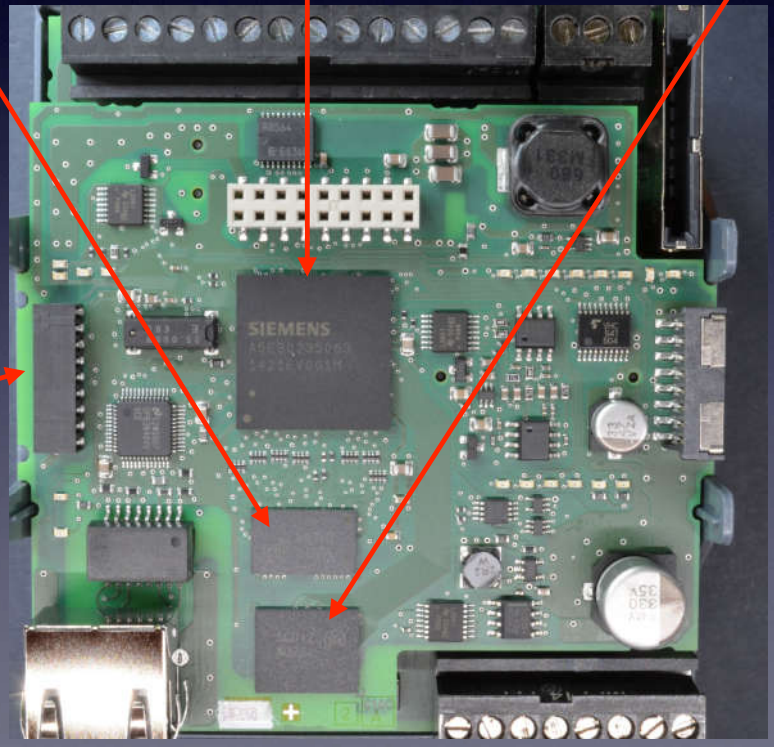
1GB Winbond
W94AD2KB LPDDR1
SDRAM

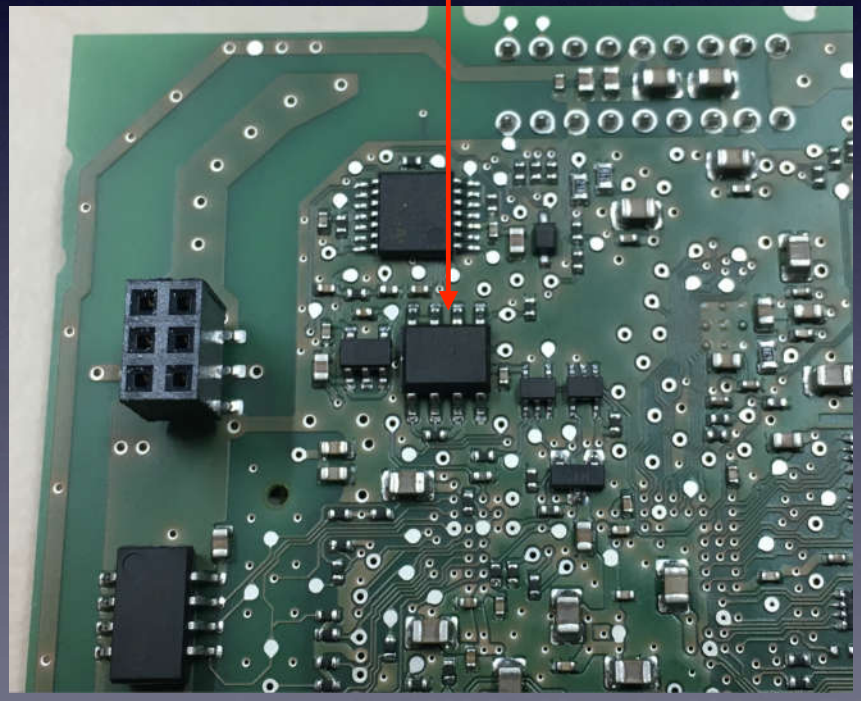
ARM CPU

128MB Micron Technologies
MT29F1G16ABBDAHC-IT:D
63-ball VFBGA NAND Flash for
Firmware.

STMicroelectronics
M25P40/MX25L4005
4MB SPI Flash for
bootloader

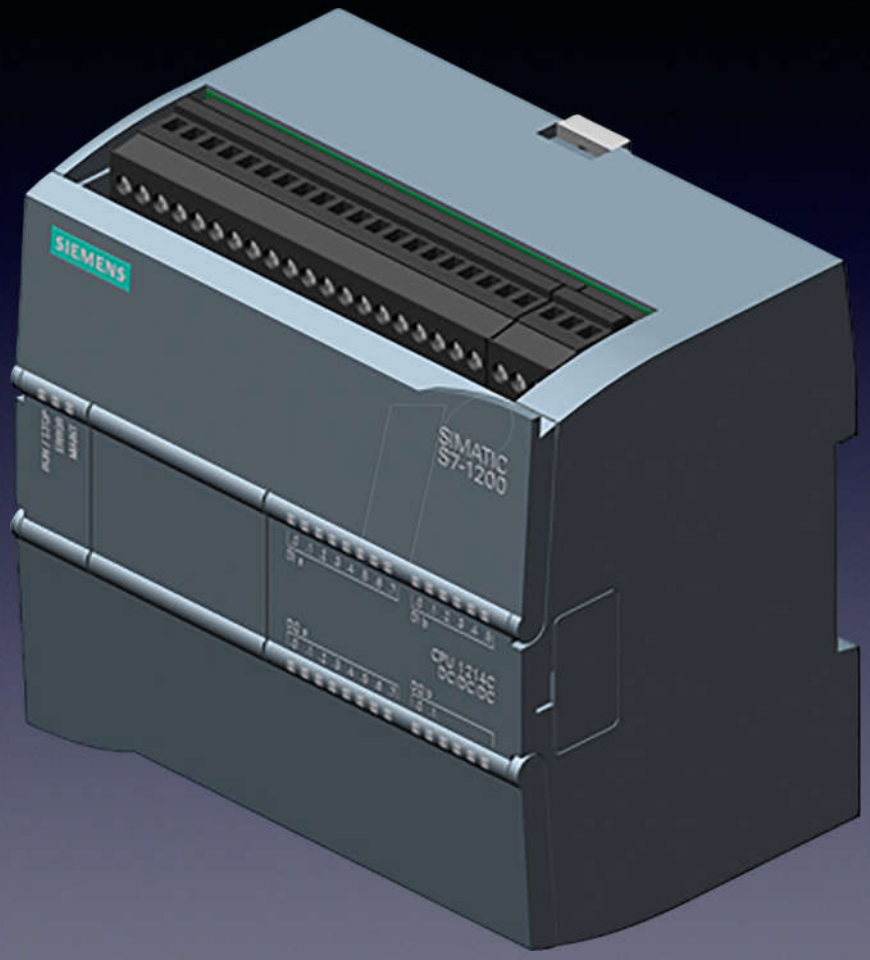
UART/Port for
RS232 extension
(CM 1241)





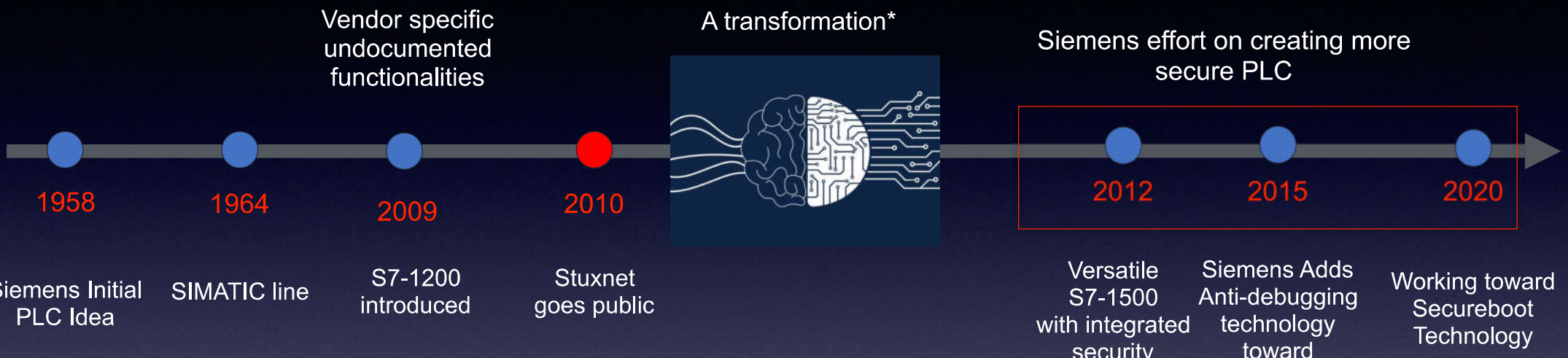
Source: http://s7detali.narod.ru/S7_1200/S7_1212C.html

S7-1200 PLC



Source: http://s7detali.narod.ru/S7_1200/S7_1212C.html

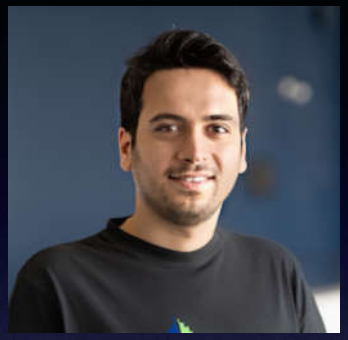
A Transformation to Hardened PLC, A Siemens Example



- In the past PLCs were not connected.
- Stuxnet pushed the vendors to bring some change.
- Vendors add security features and design their new security model
- Still have features/code which undermines the newly established security model.

*https://new.siemens.com/global/en/company/about/history/history-features/60-years-of-simatic.html

Who Are We?



Ali Abbasi
Researcher at SYSSEC
RUB

Ruhr-University Bochum (RUB), Germany

Tobias Scharnowski
PhD Student at SYSSEC
RUB

Special Access Features on Siemens S7 PLC's

Ali Abbasi, Tobias Scharnowski
Chair for Systems Security
Ruhr-University Bochum, Germany