KCon

解锁工控设备固件提取的各类方法

高 剑

+1 进阶，
护航未来

KCon

+1

Who am I

- 高剑(@ic3sw0rd)
- 工控安全研究员@绿盟科技
- 专注工业控制系统漏洞挖掘&业务系统风险评估及测试
- 看雪 SDC2020、CIS2020、HITB AMS 2021、ICS Cyber Security 2021、HITB SIN 2021、HITCON2021

KCon

+1

CONTENTS

目录

KCon

+1

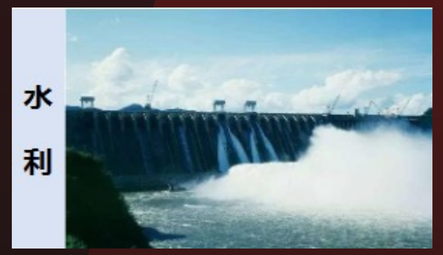
背景介绍

KCon

+1

背景介绍—工业控制系统

工业控制系统（Industrial Control Systems，ICS）：工业控制系统是一个通用术语，它包括多种工业生产中使用的控制系统，包括**监控和数据采集系统（SCADA）、分布式控制系统（DCS）和可编程逻辑控制器（PLC）**等，现已广泛应用在工业部门和关键基础设施中。----《GB/T 32919-2016 信息安全技术 工业控制系统安全控制应用指南》



国家关键信息基础设施

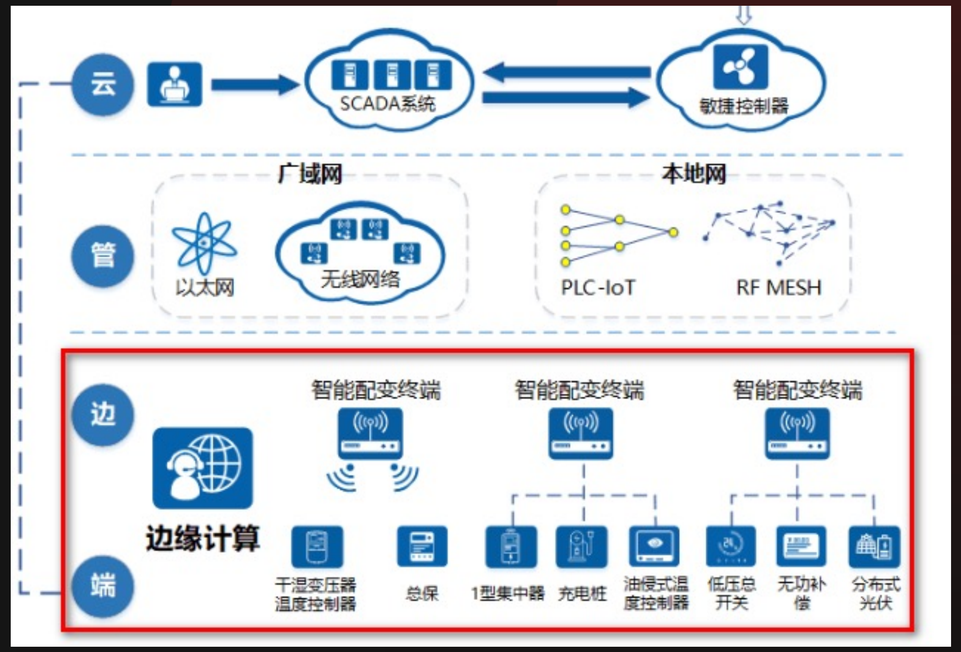
国家安全，国计民生
公共利益，社会稳定

KCon

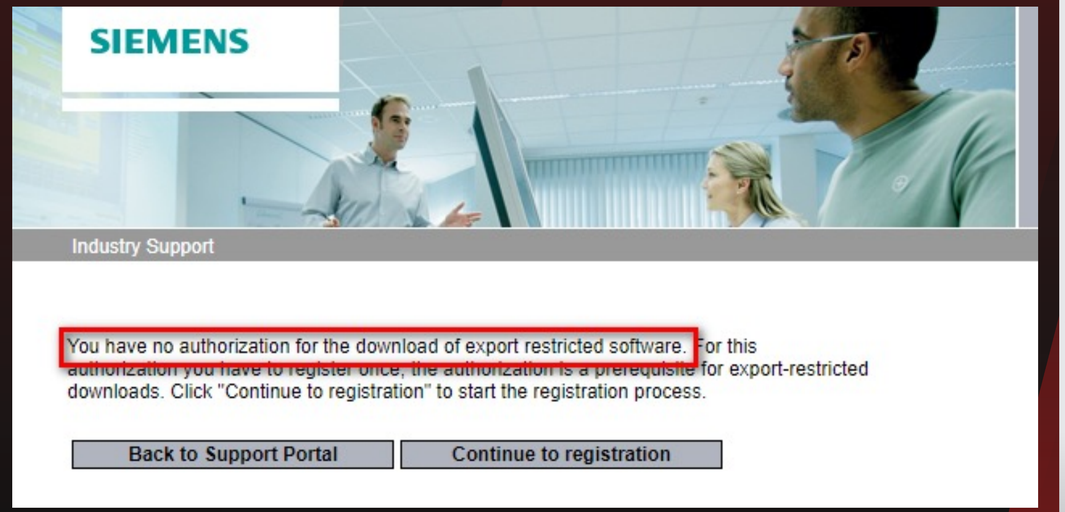
+1

背景介绍—研究方向&难点

- 传统的工控系统普渡模型中，关注**level1、level0层设备**的安全研究与测试；
- 新型的“云管边端”架构中，关注**边缘计算层、端点设备层**的各类设备安全研究与测试；
- 关注国内外工控市场份额占有率高、品牌效应强、使用广泛的设备；



- 研究对象难购买、价值高；
- **工控设备闭源程度高，相关固件、资料、软件等难以获取；**
- 硬件属性极强，采用实时操作系统或自研操作系统；
- 不可调试，没有通用工具甚至没有可调试的接口，不能定位漏洞点、不能采取更多的技术措施；
- 研究价值难以最大化，目前设备类漏洞生态未形成没有足够的动力激发更多的人加入，技术分享有限，研究不深入或者重复研究；
- ……



KCon

+1

背景介绍—工控设备固件获取的挑战

必要性：是深入研究设备各类机制及脆弱性的前提，拿到未加密、可审计的固件才能发现**更多的“秘密”**

- 大型工控厂商设备固件均经过**特殊处理**；
- 部分工控厂商不提供设备固件下载；
- 工控设备一般价值较高损坏风险较大；
- 工控设备有别于IOT类设备使用多为RTOS，无法照搬IOT类设备固件提取思路；

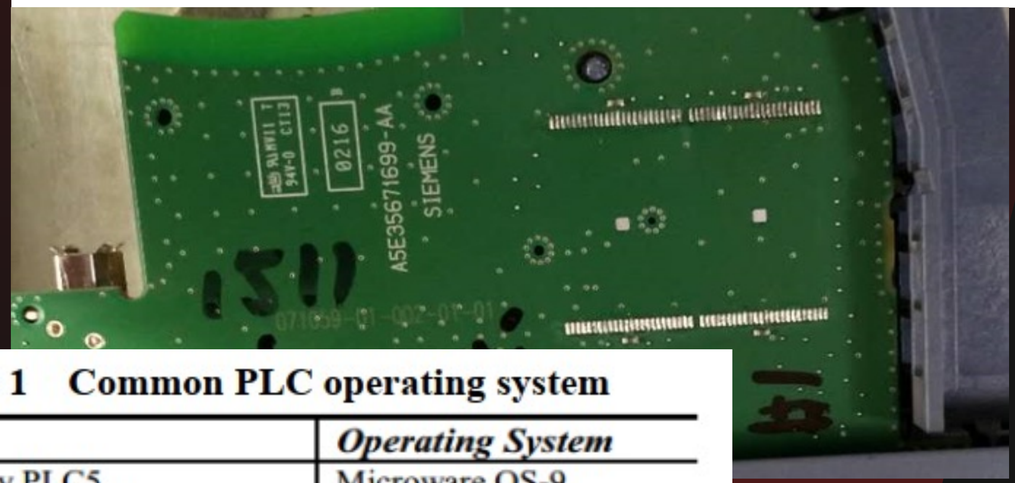
Mitsubishi Electric has fixed this vulnerability in the following products:

- MELSEC-Q Series QJ71E71-100: First five digits of serial number 24062 and later
- MELSEC-L Series LJ71E71-100: First five digits of serial number 24062 and later
- MELSEC iQ-R Series RD81MES96N: firmware Version 09 and later

For more information on how to patch individual systems, please contact Mitsubishi Electric support.

If updating to a fixed version is not possible, Mitsubishi Electric recommends users take the following mitigations to minimize risk:

- Use a firewall, virtual private network (VPN), web application firewall (WAF), etc. to prevent unauthorized access when Internet access is required.
- Use within a LAN and block access from untrusted networks and hosts through firewalls.

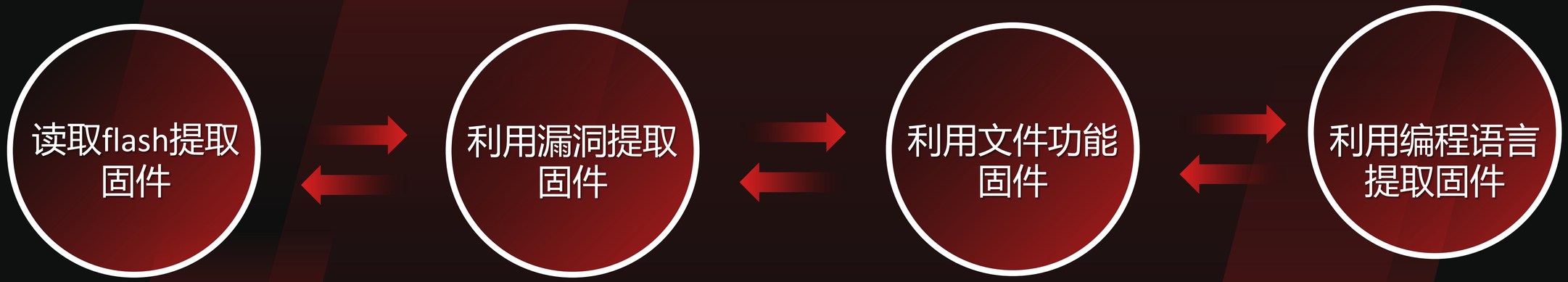


名称	修改日期	类型	大小
6ES7 511-1AK01-0AB0 V02.00.03.upd	2016/12/5 23:13	UPD 文件	18,295 KB
6ES7 215-1AG40-0XB0 V04.05.02.upd	2022/1/10 23:27	UPD 文件	15,024 KB
6ES7 288-1SR20-0AA0 V02.05.01.upd	2020/5/22 9:38	UPD 文件	1,968 KB
BMEH582040_FW_SV03.10.ldx	2019/11/19 19:46	LDX 文件	4,351 KB
BMEH582040S_FW_SV03.10.ldx	2019/11/19 19:46	LDX 文件	4,456 KB
BMEH584040_FW_SV03.10.ldx	2019/11/19 19:46	LDX 文件	4,351 KB
BMEH584040S_FW_SV03.10.ldx	2019/11/19 19:46	LDX 文件	4,456 KB

Table 1 Common PLC operating system

PLC	*Operating System*
Allen-Bradley PLC5	Microware OS-9
Allen-Bradley ControlLogix	VxWorks
Emerson DeltaV	VxWorks
Schneider Modicon Quantum	VxWorks
Yokogawa FA-M3	Linux
Wago 750	Linux
PLC reference platform	QNX Neutrino
Siemens SIMATIC WinAC RTX	Microsoft Windows

典型的几类工控设备固件提取方法


读取flash提取固件
利用漏洞提取固件
利用文件功能固件
利用编程语言提取固件

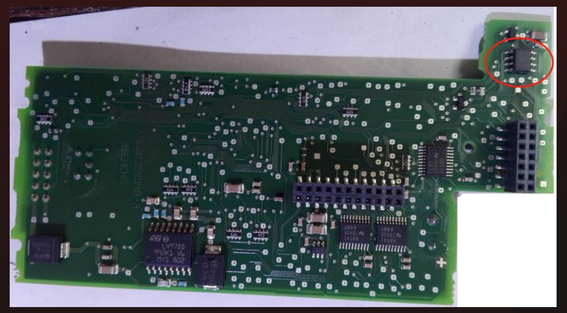
KCon

+1

读取flash提取固件方法的适用范围

为什么首先讲解此种方法？？？

- 此种方法在IOT设备提取固件中也是经常使用的通用方法；
- 该方法**最直接、效果最好、可快速**获取到设备flash中的内容；
- 适用于该方法的工控设备较多，可在适用范围内遴选合适设备采用此种方法；
- 此种方法涵盖了包括但**不限于拆焊芯片、引脚法读取芯片、测点飞线读取芯片等方法、拆焊CPU读取等**；

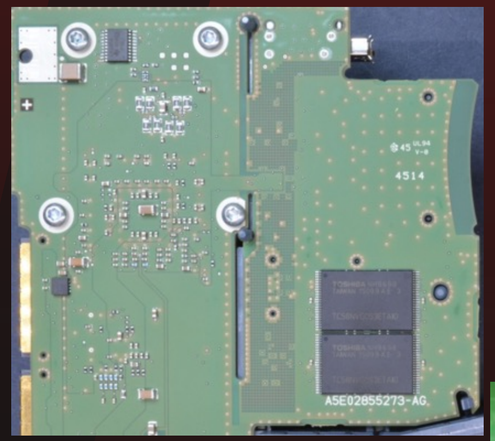


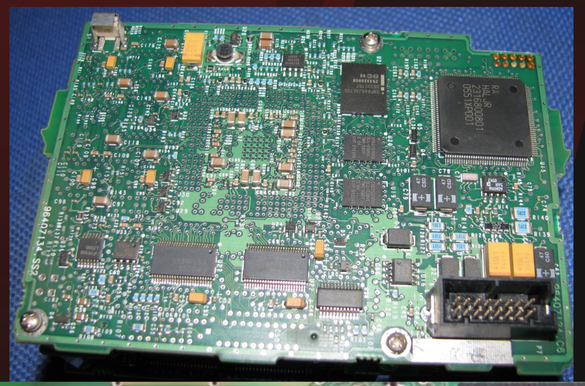
哪些工控设备可以采用该方法？？？

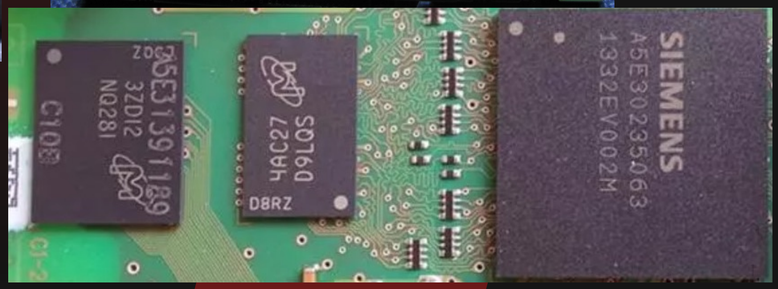
- 工控设备价值在经济能力承受范围内；
- 最好有片外flash的工控设备；
- 最好该工控设备有备品，而非独有设备，防止损毁后无法继续研究；

此种方法需要什么技能？？？

- 有基础硬件元器件、电路板知识；
- 掌握基本的硬件焊接技术，可以脱焊常见的8脚、28脚、48脚等芯片；
- 掌握编程器读取芯片技术；
- 掌握二进制文件修复与拼接等编程技术；







KCon

+1

读取flash提取固件方法的操作步骤

目标分析

- ✓ 拆开工控设备壳体，找到CPU板子，开始分析电路布局及器件；
- ✓ 找到CPU芯片，再围绕**CPU芯片寻找外围的flash芯片**、RAM芯片等；
- ✓ 根据芯片丝印查找datasheet，明确**芯片的各类参数**，比如容量大小等；

方案选择

- ✓ 根据分析的结果进行选择合适的flash内容读取方案；
- ✓ Flash芯片为片外，且引脚数较少，可以采用**引脚法**读取固件；
- ✓ Flash芯片为片外，且引脚数多于28，则采用**拆焊芯片**读取固件；
- ✓ Flash在CPU片内，可通过**JLINK读取**，连接方法有很多；

方案执行

- ✓ 引脚法读取flash需要将**线夹连接**到对应的引脚
- ✓ 拆焊方法需要理由热风枪或者烙铁逐步**脱取芯片**；
- ✓ JLINK连接读取需要**识别JTAG引脚**或者测点并对应链接到仿真器引脚；

编程器读取芯片内容

- ✓ 确认引线连接正确后探测芯片类型执行读取操作；
- ✓ 将芯片安装至对应的测试夹中，探测芯片类型执行读取操作；
- ✓ 正确连接后，利用JLINK对应的软件执行读取操作；

二进制文件修复与调整

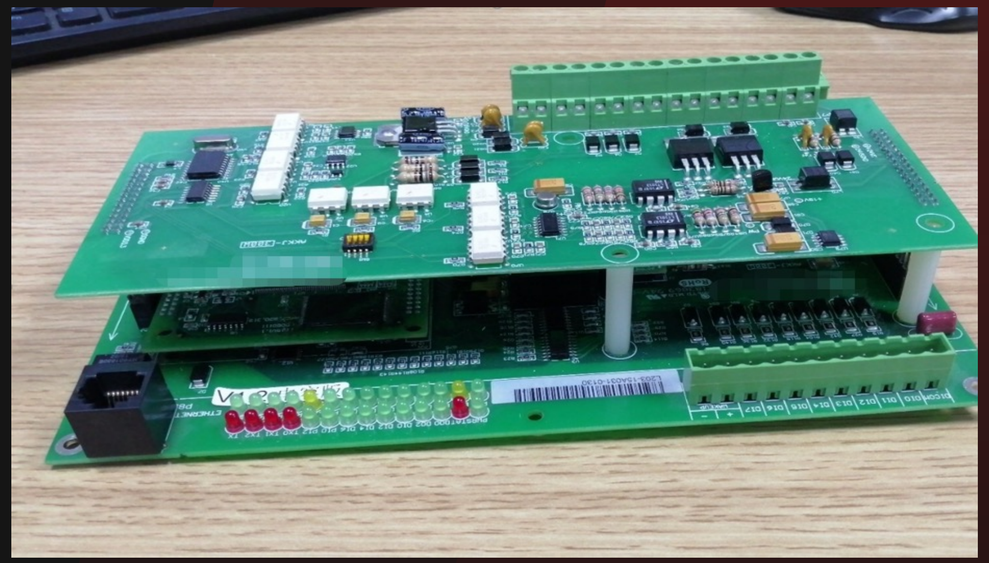
- ✓ 读取后将文件内容置于010中查看，寻找字符串信息
- ✓ 如需处理，则需对二进制文件执行**大小端调整、去除无效数据、ECC校验**等处理；

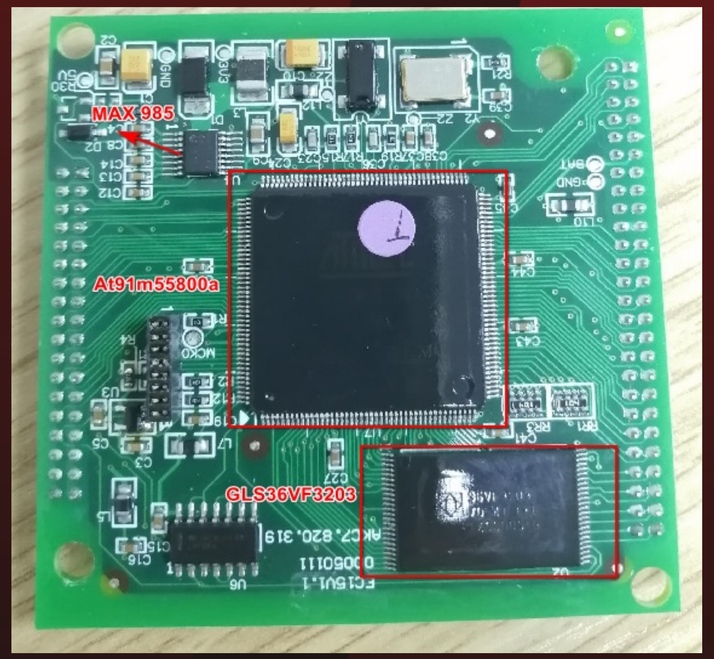
KCon

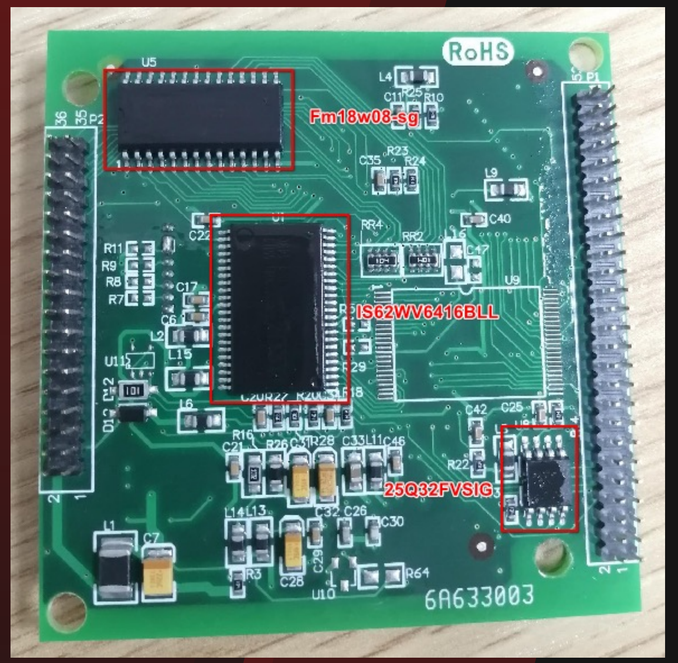
+1

读取flash提取固件方法实战案例

研究对象描述&目标分析







国内某厂商的RTU是一个一体化的智能设备，适用于SCADA系统。产品本身集电源、运算处理、通讯、IO信号采集等功能于一体。正由于信号类型丰富，功能齐全，因此在很多油气田、输油管线、水利管线上都有广泛使用。

- CPU：AT91M55800A-33AU
- FLASH：25Q32FVSIG、GLS36VF3203；
- RAM：IS62WV6416BLL、FM18W08-SG；

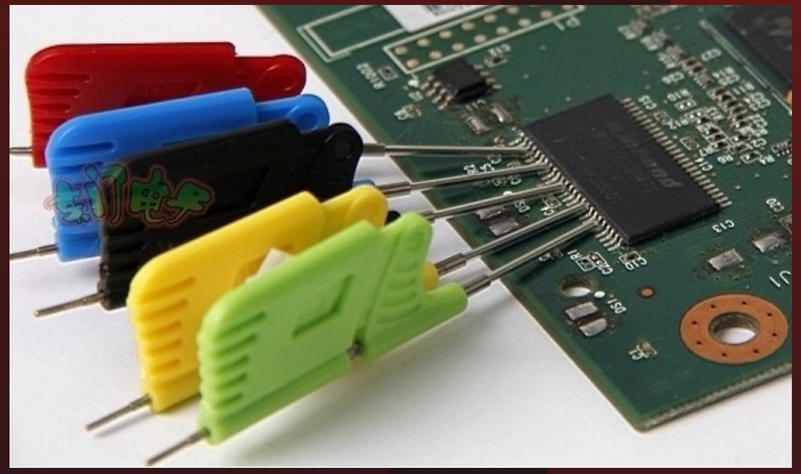
KCon

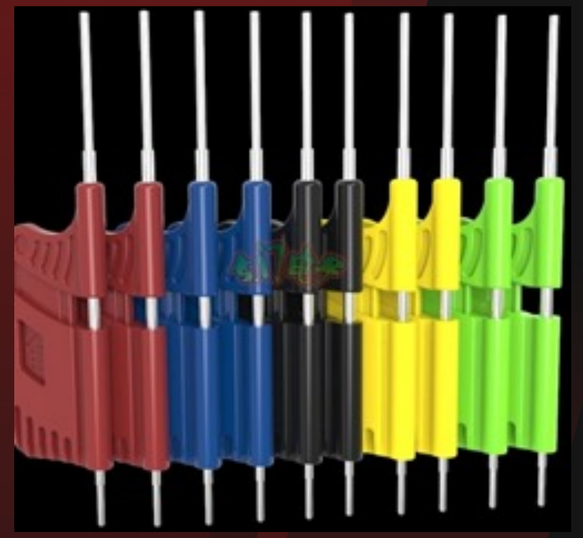
+1

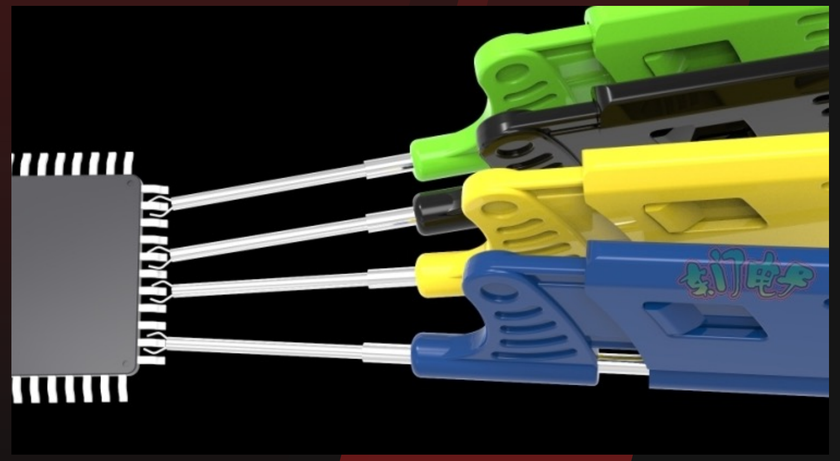
读取flash提取固件方法实战案例

方案选择

- 2片片外flash，不确定到底哪个才是目标，只能尝试各个击破；
- 25Q32FVSIG为8脚芯片，采用**引脚法，利用线夹夹住引脚读取**；
- GLS36VF3203为48脚芯片，采用**拆焊法，脱焊芯片**后读取内部数据；







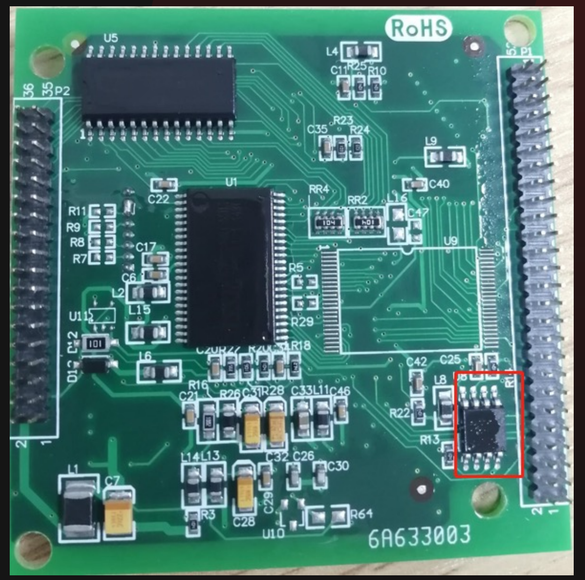
KCon

+1

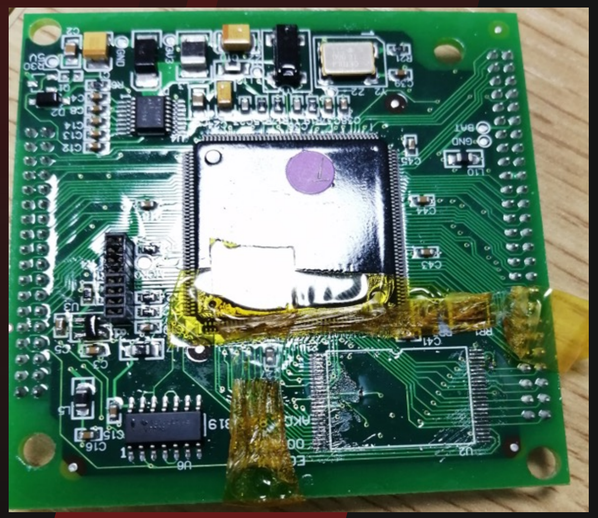
读取flash提取固件方法实战案例

方案执行

- 利用线夹的一端夹住25Q32FVSIG的8个引脚，另一端需要连接至编程器的对应位置；
- 由于8个引脚芯片间距较大，夹取比较容易，夹住后最好利用万用表量测8脚之间没有互连；



- 利用高温胶带固定GLS36VF3203芯片周围的其他器件；
- 调整热风枪至380°C左右，开始对芯片引脚部分加热，加热要均匀频繁移动热风枪；
- 加热一定时间后利用镊子轻轻夹取出芯片，然后对芯片引脚进行清洗除去残留焊锡



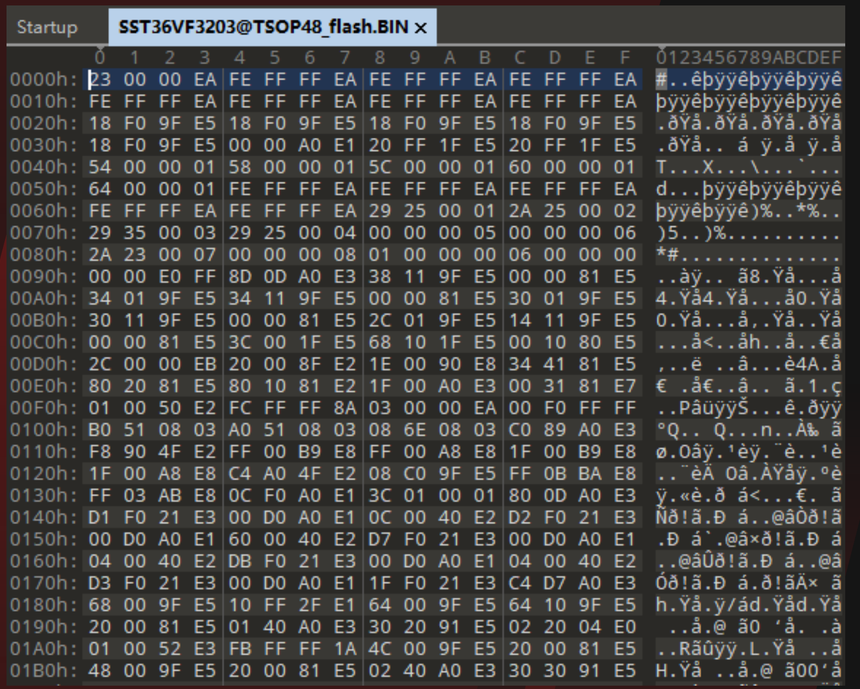
KCon

读取flash提取固件方法实战案例

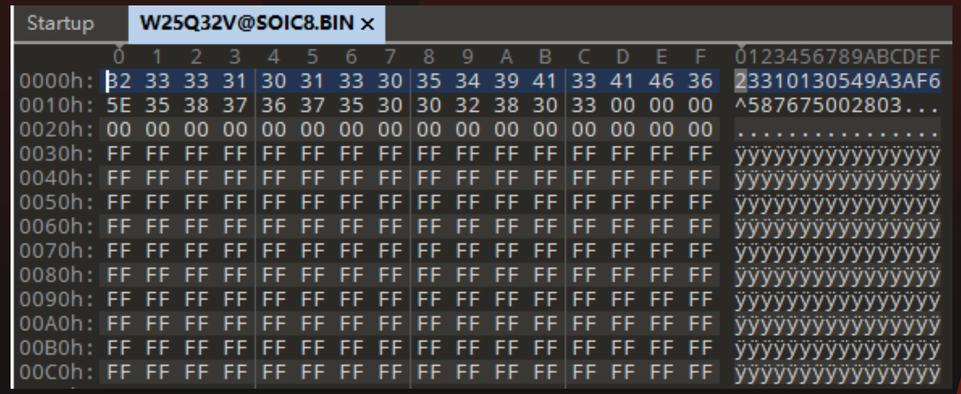
+1

编程器读取芯片内容

RT809H是一种性价比高、可靠、快速的通用编程器，在维修界使用很广泛。该编程器工作时直接与电脑USB2.0高速接口通讯，软件内置驱动程序，安装操作十分方便。编程器支持主流器件，类型包括 E/EPROM、MCU、EC、SPI NOR闪存、并行NOR闪存、SPI NAND、并行NAND、ONENAND、MCP、EMMC、EMCP 等，使用时可以为不同封装的芯片选用合适的芯片座进行连接。



GLS36VF3203芯片中内容，该芯片存储了固件内容



25Q32FVSIG芯片内容，可以看到该芯片为数据存储单元，当没有应用数据时均为FF填充

KCon

读取flash提取固件方法实战案例

二进制文件修复与调整

- 首先翻看二进制文件，根据经验判断是否存在大量不连续的段、有无明显加密特征等；
- 利用strings或者010浏览二进制文件，寻找字符串查看字符串顺序是否连续、可识别；
- 利用binwalk工具识别二进制架构等信息；

```
1E40h: 00 78 A0 E1 27 78 A0 E1 07 00 A0 E1 58 D0 8D E2  .x á'x á.. áXÐ.â
1E50h: F0 41 BD E8 1E FF 2F E1 48 61 72 64 77 61 72 65  ðA½è.ÿ/áHardware
1E60h: 20 49 6E 66 6F 72 6D 61 74 69 6F 6E 20 52 65 61   Information Rea
1E70h: 64 20 46 61 69 6C 75 72 65 21 0D 0A 00 00 00 00  d Failure!......
1E80h: 53 6E 3A 20 00 00 00 00 52 6E 3A 20 00 00 00 00  Sn: ....Rn: ....
1E90h: 0D 0A 00 00 48 6E 3A 20 00 00 00 00 45 43 6F 72  ....Hn: ....ECor
1EA0h: 65 48 61 72 64 77 61 72 65 3A 20 00 08 6F 08 03  eHardware: ..o..
1EB0h: 50 5F 44 6F 77 6E 4C 6F 61 64 69 6E 67 3A 20 00  P_DownLoading: .
1EC0h: 48 6F 08 03 48 5F 49 6E 66 6F 72 6D 61 74 69 6F  Ho..H_Informatio
1ED0h: 6E 3A 20 00 50 5F 49 6E 66 6F 72 6D 61 74 69 6F  n: .P_Informatio
1EE0h: 6E 3A 20 00 F0 41 2D E9 30 D0 4D E2 00 70 A0 E1  n: .ðA-é0ÐMâ.p á
1EF0h: 01 50 A0 E1 00 40 A0 E3 30 00 54 E3 07 00 00 2A  .P á.@ ã0.Tã...*
1F00h: 02 00 00 EA 01 00 84 E2 40 4B C0 E3 F9 FF FF EA  ...ê...„â@KÀãùÿÿê
1F10h: 04 00 87 E0 01 00 D0 E5 04 00 CD E7 F8 FF FF EA  ..‡à..Ðå..Íçøÿÿê
1F20h: 30 20 A0 E3 00 10 A0 E3 0D 00 A0 E1 61 FA FF EB  0  ã.. ã.. áaúÿë
1F30h: 00 60 A0 E1 00 00 56 E3 03 00 00 0A 34 10 8F E2  .` á..Vã....4..â
1F40h: 05 00 A0 E1 6D 14 00 EB 02 00 00 EA 48 10 8F E2  .. ám..ë...êH..â
1F50h: 05 00 A0 E1 69 14 00 EB 05 00 A0 E1 6D 14 00 EB  .. ái..ë.. ám..ë
1F60h: 00 88 A0 E1 28 88 A0 E1 08 00 A0 E1 30 D0 8D E2  .ˆ á(ˆ á.. áÐ.â
1F70h: F0 41 BD E8 1E FF 2F E1 4D 6F 64 75 6C 65 20 49  ðA½è.ÿ/áModule I
1F80h: 6E 66 6F 72 6D 61 74 69 6F 6E 20 57 72 69 74 65  nformation Write
1F90h: 20 46 61 69 6C 75 72 65 21 0D 0A 00 4D 6F 64 75   Failure!...Modu
1FA0h: 6C 65 20 49 6E 66 6F 72 6D 61 74 69 6F 6E 20 57  le Information W
```

```
root@kali:~# binwalk -Y SST36VF3203@TSOP48_flash.BIN

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
1400          0x578           ARM executable code, 32-bit, little endian, at least 882 valid instructions
```

```
0720h: 80 E5 94 0A 9F E5 88 1A 9F E5 00 10 80 E5 8C 0A  €å”.Yå^.Yå..€åŒ.
0730h: 9F E5 7C 1A 9F E5 00 10 80 E5 84 0A 9F E5 70 1A  Yå|.Yå..€å„.Yåp.
0740h: 9F E5 00 10 80 E5 7C 0A 9F E5 64 1A 9F E5 00 10  Yå..€å|.Yåd.Yå..
0750h: 80 E5 74 0A 9F E5 58 1A 9F E5 00 10 80 E5 6C 0A  €åt.YåX.Yå..€ål.
0760h: 9F E5 4C 1A 9F E5 00 10 80 E5 64 0A 9F E5 00 10  YåL.Yå..€åd.Yå..
0770h: A0 E3 00 10 80 E5 5C 0A 9F E5 00 10 A0 E3 00 10   ã..€å\.Yå.. ã..
0780h: 80 E5 54 0A 9F E5 00 10 A0 E3 00 10 80 E5 4C 0A  €åT.Yå.. ã..€åL.
0790h: 9F E5 00 10 A0 E3 00 10 80 E5 44 0A 9F E5 00 10  Yå. ã..€åD.Yå..
07A0h: A0 E3 00 10 80 E5 3C 0A 9F E5 00 10 A0 E3 00 10   ã..€å<.Yå.. ã..
07B0h: 80 E5 34 0A 9F E5 00 10 A0 E3 00 10 80 E5 2C 0A  €å4.Yå.. ã..€å,.
07C0h: 9F E5 00 10 A0 E3 00 10 80 E5 24 0A 9F E5 24 1A  Yå. ã..€å$.Yå$.
07D0h: 9F E5 00 10 80 E5 20 0A 9F E5 18 1A 9F E5 00 10  Yå..€å .Yå..Yå..
07E0h: 80 E5 18 0A 9F E5 0C 1A 9F E5 00 10 80 E5 10 0A  €å..Yå..Yå..€å..
07F0h: 9F E5 00 1A 9F E5 00 10 80 E5 08 0A 9F E5 F4 19  Yå..Yå..€å..Yåô.
0800h: FF FF 9F E5 00 10 80 E5 00 0A 9F E5 00 1A 9F E5  ÿÿYå..€å..Yå..Yå
0810h: AE 6A 6B 62 AB C2 FB 93 AF 34 88 A1 7F 89 FF FF  ®jkb«Âû¯4ˆ¡.‰ÿÿ
0820h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
0830h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
0840h: 00 10 80 E5 EC 09 9F E5 F4 19 9F E5 00 10 80 E5  ..€åì.Yåô.Yå..€å
0850h: F4 09 9F E5 E8 19 9F E5 00 10 80 E5 EC 09 9F E5  ô.Yåè.Yå..€åì.Yå
0860h: DC 19 9F E5 00 10 80 E5 E4 09 9F E5 D0 19 9F E5  Ü.Yå..€åä.Yåð.Yå
```

```
1FF0h: 00 60 00 00 00 60 00 00 00 60 00 00 00 60 00 00  .`...`...`...`..
2000h: 32 56 30 2E 32 2E 32 2C 31 30 2D 30 33 30 32 2D  2V0.2.2,10-0302-
2010h: 20 36 42 28 69 75 64 6C 37 3A 32 39 2C 33 32 31   6B(iudl7:29,321
2020h: 32 3A 3A 38 35 35 52 2C 6C 65 00 29 00 00 00 00  2::855R,le.)....
2030h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 40 00  ..............@.
2040h: 40 02 20 00 00 4C 00 40 6C 20 00 01 20 7C 7C 31  @. ..L.@l .. ||1
2050h: 45 50 4D 52 4E 41 4E 45 5F 54 41 52 54 4D 53 45  EPMRNANE_TARTMSE
2060h: 09 54 20 7C 30 20 20 20 00 7C 00 00 20 7C 7C 32  .T |0   .|.. ||2
2070h: 45 4E 5F 57 4F 43 50 55 45 4C 5F 52 49 54 49 4D  EN_WOCPUEL_RITIM
2080h: 47 4E 7C 09 20 20 20 30 7C 20 00 00 00 00 00 00  GN|.   0| ......
2090h: 21 94 F0 FF E1 93 0C 00 E0 3F 01 00 FF 63 A0 86  !”ðÿá“..à?..ÿc †
20A0h: 00 60 00 00 FF 37 FF FF 82 40 F8 FF E1 83 0C 00  .`..ÿ7ÿÿ‚@øÿáƒ..
20B0h: 21 38 10 00 80 4E 20 00 00 00 00 00 00 00 00 00  !8..€N .........
20C0h: 08 7C A6 02 21 94 E0 FF A1 93 14 00 C1 93 18 00  .|¦.!”àÿ¡“..Á“..
20D0h: 01 90 24 00 E1 93 1C 00 8D 80 1C 82 E0 3F 00 FF  ..$.á“...€.‚à?.ÿ
20E0h: 44 81 07 00 A0 3B 00 00 5F 91 00 00 3F 81 00 00  D... ;.._‘..?...
20F0h: 40 39 0F 0F 29 65 01 00 3F 91 00 00 BF 93 74 01  @9..)e..?‘..¿“t.
2100h: BF B3 52 09 5F B1 50 09 BF 93 BC 0A 20 3D 02 00  ¿³R._±P.¿“¼. =..
2110h: 29 61 30 FF 3F 91 B8 0A BF B3 62 09 80 39 F2 00  )a0ÿ?‘¸.¿³b.€9ò.
2120h: 9F B1 60 09 C0 3F CC 55 DE 63 33 AA DF 93 80 03  Ÿ±`.À?ÌUÞc3ªß“€.
2130h: DF 93 84 03 64 80 13 00 01 48 B9 AC 2D 81 1C 82  ß“„.d€...H¹¬-.‚
```

2KB page size

8bit ECC 模式 2KB page size 配置下，驱动软件中的数据结构如图 4-19 所示。

图4-19 8bit ECC 模式 2KB page size 配置下驱动软件中的数据结构

DATA(2048B)	BB	CTRL	RSV

BB：bad block，2byte
CTRL：留给软件用的控制区域，30byte

写到 NAND Flash 中的数据结构如图 4-20 所示。把软件有效数据切成 2 个 1040byte 的数据块，每 1040byte 数据计算一次 ECC。写到 NAND Flash 器件内部的数据结构会自动调整为 1040byte 数据+14byte ECC 码的格式交替存放，共 2 组，BB 信息放在 NAND Flash spare 区的首两个字节。

图4-20 8bit ECC 模式 2KB page size 配置下 NAND Flash 中数据结构

DATA(1040)+ECC(14)+DATA(994)+BB(2) +DATA(14)+ECC(14)+CTRL(30)

利用设备漏洞提取固件-简介

什么样的漏洞可以用来提取固件？

可以直接或者通过exploit间接访问物理内存的漏洞都可以用来提取固件

- **漏洞可以修复：该种漏洞存在于固件中，通过发布新版本修复；**
- **漏洞不可完全修复：该种漏洞存在于硬件中，不能彻底修复或者不能在历史版本上修复；**

- **回退至有漏洞的固件版本，在该版本上提取固件；**
- **寻找有漏洞的硬件版本，在该特定版本的设备上提取固件，即使固件再升级也不影响；**

CVE-2020-15782

CWE-119 Improper Restriction of Operations within the Bounds of a Memory Buffer

CVSS v3.1 Score: 8.1

Affected devices are vulnerable to a memory protection bypass through a specific operation. A remote unauthenticated attacker with network access to TCP port 102 could potentially write arbitrary data and code to protected memory areas or read sensitive data to launch further attacks.

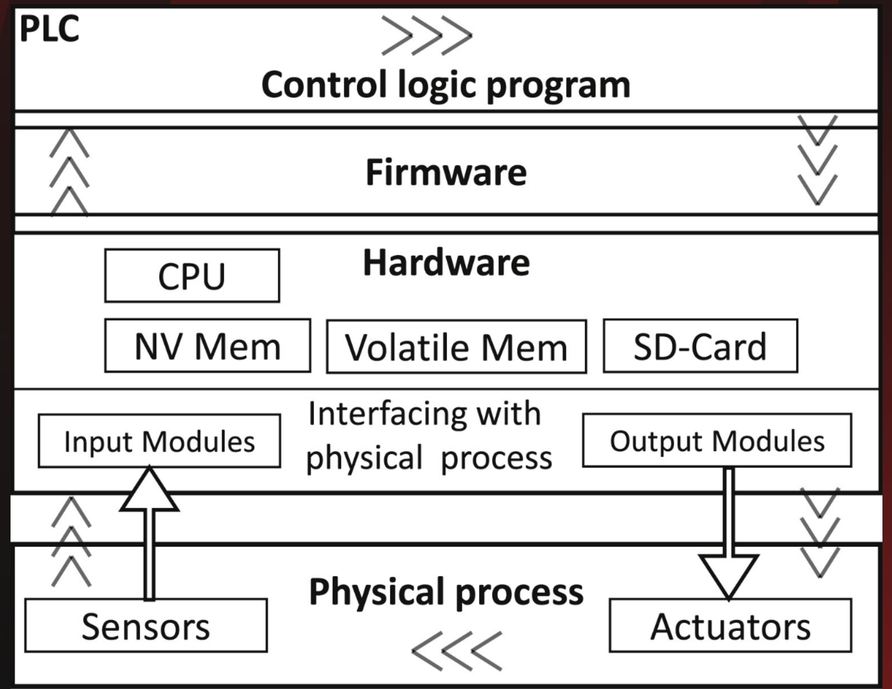
Today, Siemens released advisory SSA-434534 that notifies its users of this vulnerability. Siemens also released updates for various products, including S7-1500 and S7-1200, that remediate the vulnerability. Users are urged to update to the latest versions. The company said it is preparing further updates for products where updates are not yet available; Siemens also pr

PLC memory address

Definition

The appropriate product category ▸ Programmable Controllers

A physical address in RAM in the CPU Unit. Setting a PLC memory address in an index register (IR0 to IR15) enables indirect addressing of I/O memory. It is different form an address in the I/O memory (user addresses by area, such as D100 or W100) that is specified for an instruction operand.



KCon

+1

利用设备漏洞提取固件-施耐德M580

M580协议漏洞读取内存数据

```
205     case 0x26u:
206       pu_DataDictionary(a1, v59, a3, a4);
207       return (_BYTE *)exh_EndTry(v61, &v68);
208     case 0x27u:
209       pu_DataDictionaryPreload(a1, v59, a3, a4);
210       return (_BYTE *)exh_EndTry(v61, &v68);
211     case 0x28u:
212       pu_ReadPhysicalAddress(a1, v59, a3, a4);
213       return (_BYTE *)exh_EndTry(v61, &v68);
214     case 0x29u:
215       if ( a5 != 2 )
216         pumem_error(1200906, -32638, 0);
217       pu_WritePhysicalAddress(a1, v59, a3, a4);
218       return (_BYTE *)exh_EndTry(v61, &v68);
219     case 0x2Au:
220       pu_BrowseEvents(a1, v59, a3, a4);
221       return (_BYTE *)exh_EndTry(v61, &v68);
```

```
1  int __fastcall pu_ReadPhysicalAddress(unsigned __int8 *a1, unsigned __int16 a2, _BYTE *a3, _WORD *a4)
2  {
3    int v7; // r6
4    int v8; // r8
5    int (__fastcall *v9)(int, int, _DWORD); // r3
6    int result; // r0
7    __int16 v11; // r3
8
9    v7 = a1[5] | (a1[6] << 8);
10   v8 = a1[1] | (a1[2] << 8) | (a1[3] << 16) | (a1[4] << 24);
11   if ( a2 > 6u )
12   {
13     if ( (a1[5] | (a1[6] << 8)) + 3 > (unsigned int)(unsigned __int16)*a4 )
14       goto LABEL_8;
15 LABEL_4:
16     v9 = (int (__fastcall *)(int, int, _DWORD))*off_11DA84;
17     if ( !*off_11DA84 )
18       goto LABEL_9;
19     goto LABEL_5;
20   }
21   pumem_error(0x129F3A, 0xFFFF8088, 0);
22   if ( (a1[5] | (a1[6] << 8)) + 3 <= (unsigned int)(unsigned __int16)*a4 )
23     goto LABEL_4;
24 LABEL_8:
25   pumem_error(0x129F3F, 0xFFFF8088, v7);
26   v9 = (int (__fastcall *)(int, int, _DWORD))*off_11DA84;
27   if ( !*off_11DA84 )
28     goto LABEL_9;
29 LABEL_5:
30   if ( v9(v8, v7, 0) != 1 )
31     return pumem_error(0x129F50, 0xFFFF9B90, a1[5] | (a1[6] << 8));
32 LABEL_9:
33   result = memcpy_s(a3 + 3, (unsigned __int16)*a4, v8, a1[5] | (a1[6] << 8));
34   *a3 = -2;
35   v11 = a1[5] | (a1[6] << 8);
36   a3[1] = a1[5];
37   a3[2] = HIBYTE(v11);
38   *a4 = v11 + 3;
39   return result;
40 }
```

- 0x28为读取物理地址的功能码，该功能码的执行不需要授权即可进行；
- 读取物理地址作为未文档化的功能，没有加以限制更没有对地址区域进行限制可读取内存数据；
- 构造符合0x28功能码结构的请求报文，不断变化读取地址和读取size即可对内存数据全部dump;

0x5A	Session key	0x28	address	size
			4字节	2字节

KCon

+1

利用设备漏洞提取固件-西门子S7-1200

S7-1200系列PLC特殊访问漏洞

- 该漏洞存在时间很久，从2013年就开始携带该漏洞；
- 该漏洞的存在是为了方便工厂测试、研发人员定位或者诊断问题使用；
- PLC上的URAT特殊访问功能未文档化，是一个隐藏功能；
- UART的特殊访问功能实现代码在PLC的uboot芯片中，在出厂时就已经焊接上且生命周期中不支持升级；
- 该漏洞不仅存在于S7-1200系列PLC，通用存在于S7-200 SMART系列PLC
- 由于该功能出厂携带不可升级，因此市场上大批量服役PLC都受影响，且不可能完全修复；
- 该漏洞不可远程执行，只能物理接触PLC后才能触发，因此CVE评分不会很高；

Vulnerability CVE-2019-13945

There is an access mode used during manufacturing of the affected devices that allows additional diagnostic functionality.

The security vulnerability could be exploited by an attacker with physical access to the UART interface during boot process.

CVSS v3.1 Base Score	6.8
CVSS Vector	CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H/E:P/RL:U/RC:C
CWE	CWE-749: Exposed Dangerous Method or Function

Affected Product and Versions	Remediation
SIMATIC S7-1200 CPU family V4.x (incl. SIPLUS variants): All versions with Function State (FS) < 11	Update to version >= V4.4.1 and Function State (FS) >= 11
SIMATIC S7-1200 CPU family < V4.x (incl. SIPLUS variants): All versions	Firmware versions less than V4.x cannot be updated. For remediation see the recommendations from section “Workarounds and Mitigations”.
SIMATIC S7-200 SMART CPU ST20 (6ES7 288-1ST20-0AA0): All versions <= V2.5.0 and Function State (FS) <= 9	Update to version >= V2.5.1 and the latest boot loader version
SIMATIC S7-200 SMART CPU ST30 (6ES7 288-1ST30-0AA0): All versions <= V2.5.0 and Function State (FS) <= 9	Update to version >= V2.5.1 and the latest boot loader version

KCon

利用设备漏洞提取固件-西门子S7-1200

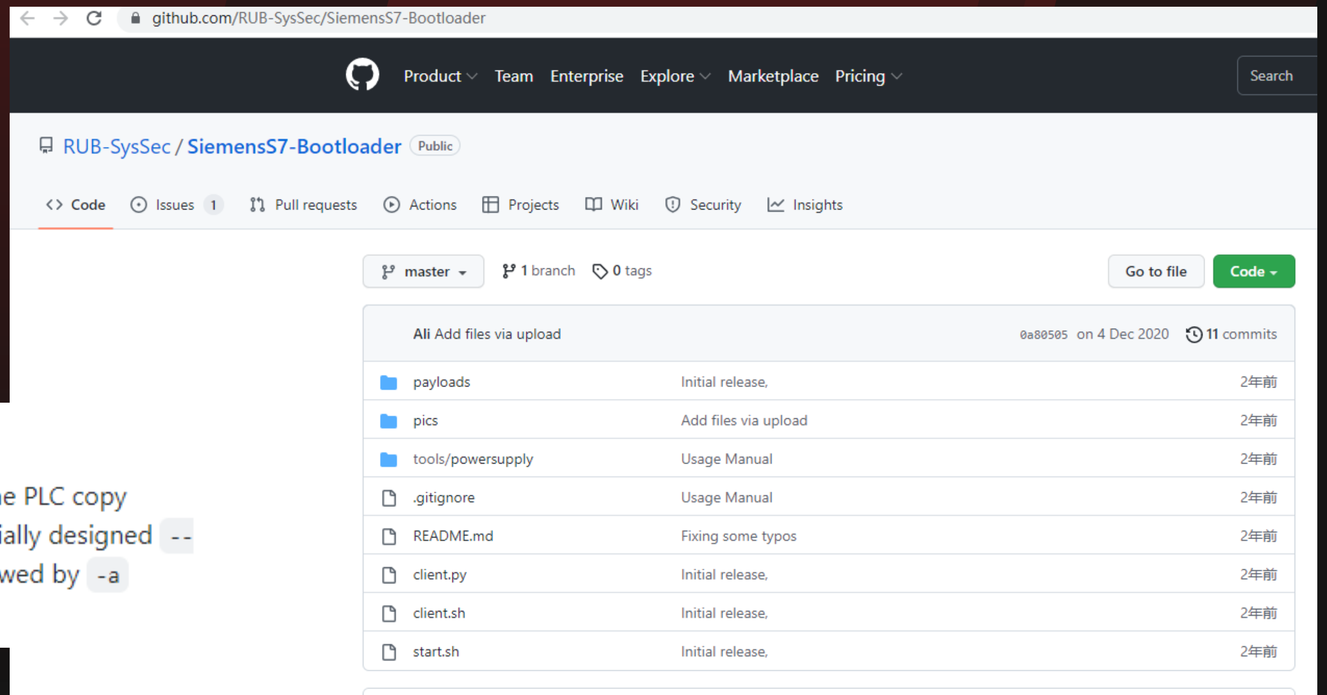
如何利用漏洞？？？

漏洞可实现的功能：

- **执行任意代码；**
- **内存取证，从内存中寻找恶意文件；**
- **固件dump,转储后可分析固件配合fuzzing技术进行漏洞挖掘；**
- **整个内存区域数据的转储，寻找关键信息；**
- **恶意利用可写入非法代码和数据；**

我们关注如何dump出内存，利用其进行固件分析

https://github.com/RUB-SysSec/SiemensS7-Bootloader



DEMO 4: Dumping S7 PLC RAM

To dump the PLC memory, we would recommend to first turn on the PLC for few seconds, to let the PLC copy contents of the NAND flash to the RAM (alternatively you can wait as long as you want!). We specially designed `--powersupply-delay` argument in our utility for this purpose. We use dump mode in our utility followed by `-a` argument which user supplies address to dump and `-l` argument for the byte size.

KCon

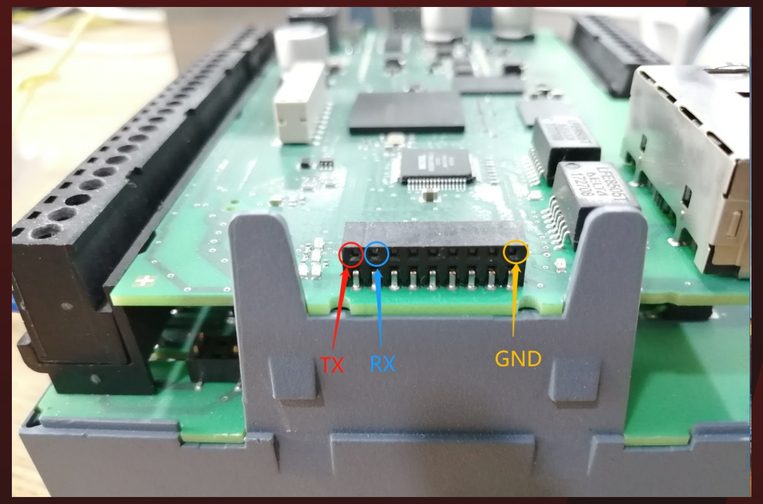
+1

利用设备漏洞提取固件-西门子S7-1200

Dump内存的环境搭建与执行

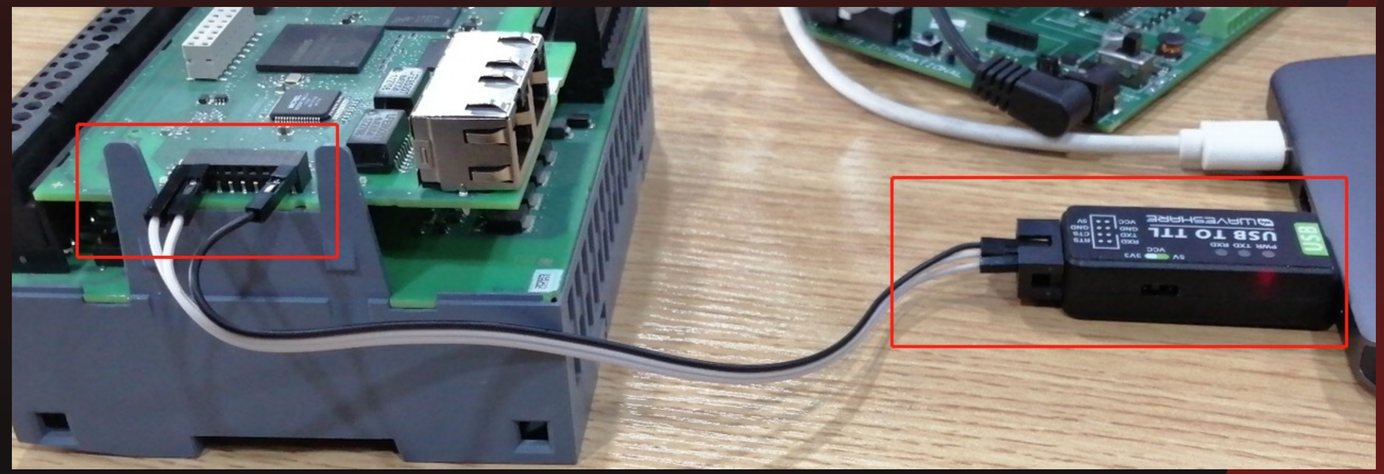
Memory Dump Hardware Tool

Memory Dump Client



环境搭建：

- **分为client端软件环境和PLC设备URAT接口、电源控制模块几个部分；**
- **UART接口在靠近S7-1200系列PLC网口的一侧；**
- **电源控制模块控制程序要能和client端程序集成；**
- **Client端软件环境可以直接使用现有基于Linux的串口转网络数据流也可以直接重新编写；**



KCon

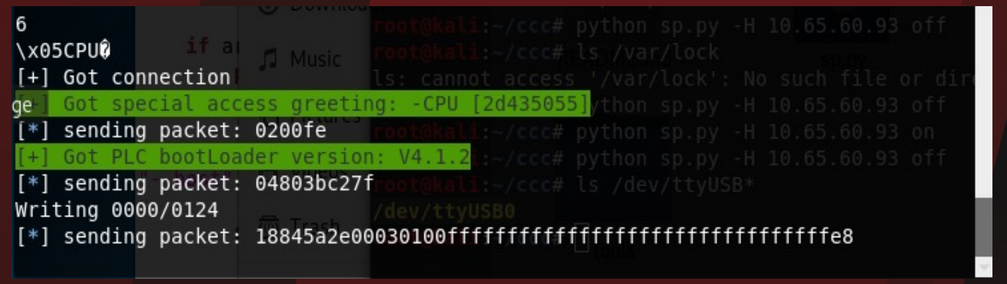
利用设备漏洞提取固件-西门子S7-1200

遇到问题及解决办法

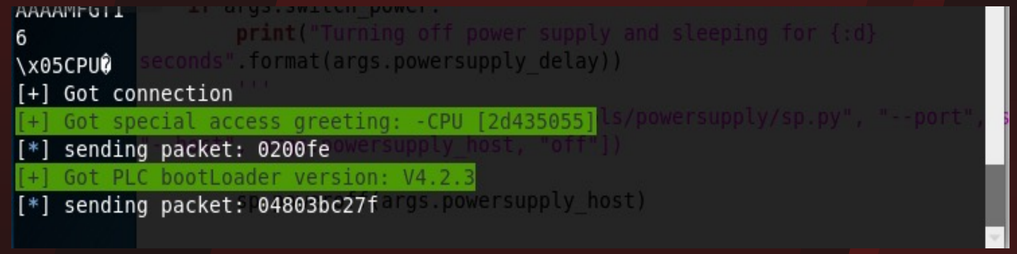
1 在使用现有client的前提下，如何选取合适的PLC设备？

- 首先确认client支持的BootLoader版本为4.2.1；
- BootLoader版本和S7-1200固件版本不一样；
- BootLoader版本和S7-1200系列PLC的FS版本有关系；
- 目前经验为：S7-1215c系列的FS为6的BootLoader版本符合client的约定；
- 如果BootLoader不是V4.2.1，则需要修改涉及到接口函数的对应硬编码地址；

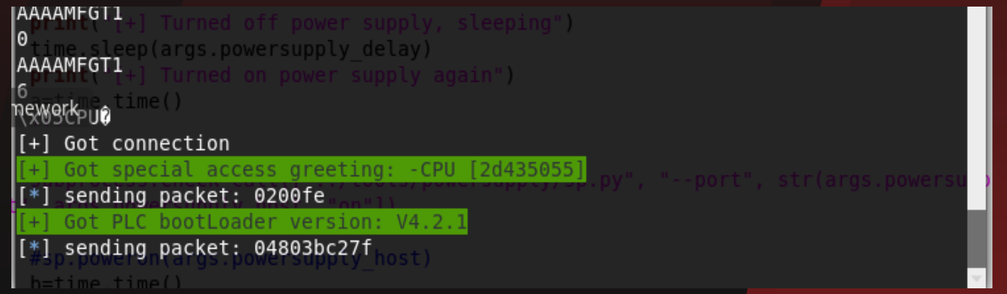
S7-1214C FS=4的BootLoader版本为V4.1.2



S7-1212C FS=10的BootLoader版本为V4.2.3



S7-1215C FS=6的BootLoader版本为V4.2.1



KCon

利用设备漏洞提取固件-西门子S7-1200

遇到问题及解决办法

2 在Linux环境下无法收到底层串口的数据导致程序处理不正常，如何解决？

- 排查问题发现，很有可能是Linux环境下USB2TTL程序中串口转网口底层代码出现问题；
- 没有直接修复该问题，而是另辟蹊径，直接在Windows下重写串口程序remote进行数据收发即可解决该问题；

```
args = parser.parse_args()

# We are currently using pwntools for the connection as those
# proved to be reliable. We may want to refactor this.
s = remote('com6') #UART TTL COM
```

```
class remote():
    def __init__(self,com):
        self.com = com
        self.ser=serial.Serial(port=com, baudrate=38400, bytesize=8, parity='E', stopbits=1, timeout=0.3)
    def send(self,data):
        #hexdump(data)
        self.ser.write(data)
    def recv(self,size,timeout=0.3,mode=1):
        if mode ==0:
            data=self.ser.read(size)
            #hexdump(data)
            return data
        else:
            while True:
                count = self.ser.inWaiting()
                if count >= size :
                    data = self.ser.read(size)
                    if data != None and len(data)>0:
                        #hexdump(data)
                        return data
                s.send(pad+magic)

                answ = s.recv(256, timeout=0.3,mode=0)
                hexdump(answ)

                if len(answ) > 3:
                    #print(answ)
                    if not answ.startswith("\5-CPU"):
                        answ += s.recv(256)
                        hexdump(answ)
                    assert(answ.startswith("\5-CPU"))
                    #s.unrecv(answ)

                    handle_conn(s, args.action, args)
                    break

    print("Done.")
```

KCon

+1

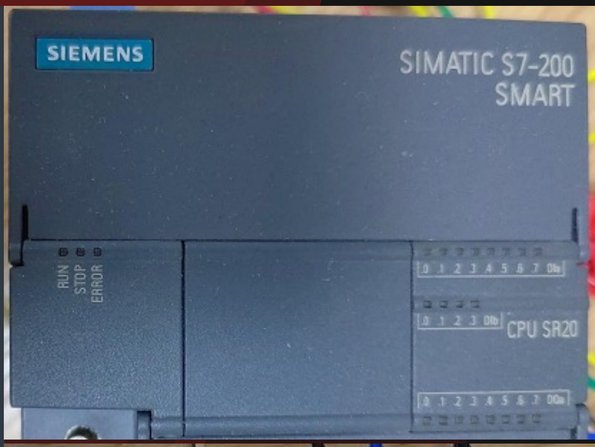
利用设备漏洞提取固件-西门子S7-1200

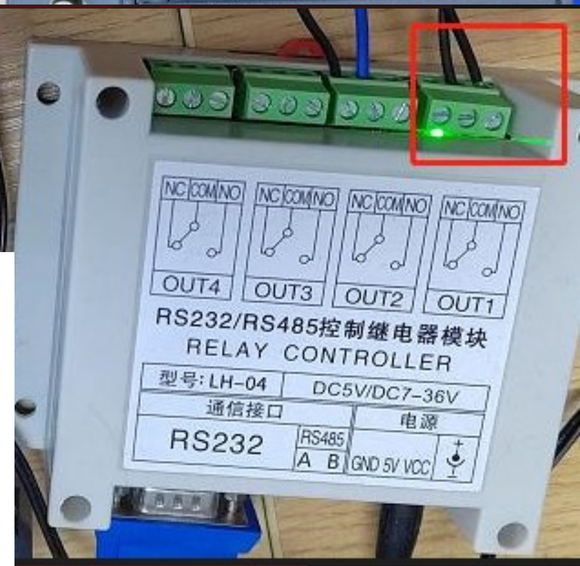
遇到问题及解决办法

3 PLC设备电源控制起什么作用？上电后多久需要断电重启？怎么控制？

- 在dump内存的环境中电源控制主要是让flash中的内容能全部加载到RAM中，然后再利用该漏洞从RAM中读取内容；
- 第一次断电上电是为了让flash拷贝至RAM，上电时间建议为30s；
- 第二次断电上电是为了进入UART的特殊访问模式来利用这个漏洞dump内存，断电时长建议为250ms左右，后续长期上电；
- 控制断电上电的方式有很多，只要是带有继电器的任何可编程可控制的模块均可；

```
if args.switch_power:
    pc=powercontrol('com4') #control Power COM

    pc.poweroff()
    print("[+] Turned off power supply, sleeping")

    time.sleep(1)

    pc.poweron()
    print("[+] Turned on power supply again")

    print args.powersupply_delay

    time.sleep(args.powersupply_delay)

    print('supply_delay------------')

    pc.poweroff()

    time.sleep(0.25)

    pc.poweron()

    print("[+] Successfully turned on power supply")
```





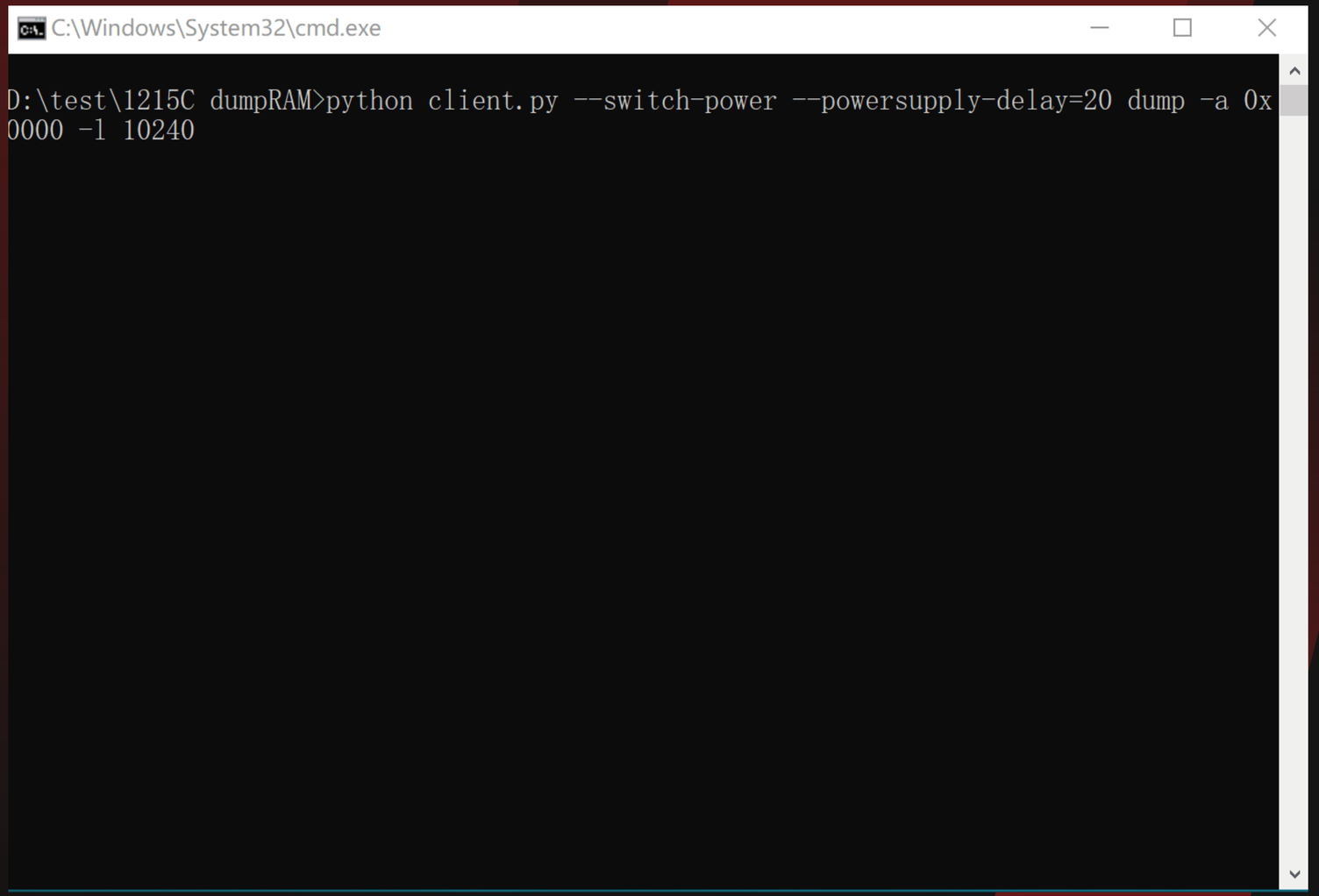
KCon

利用设备漏洞提取固件-西门子S7-1200

python client.py --switch-power --powersupply-delay=20 dump -a 0x00 -l 0x08000000

Start address

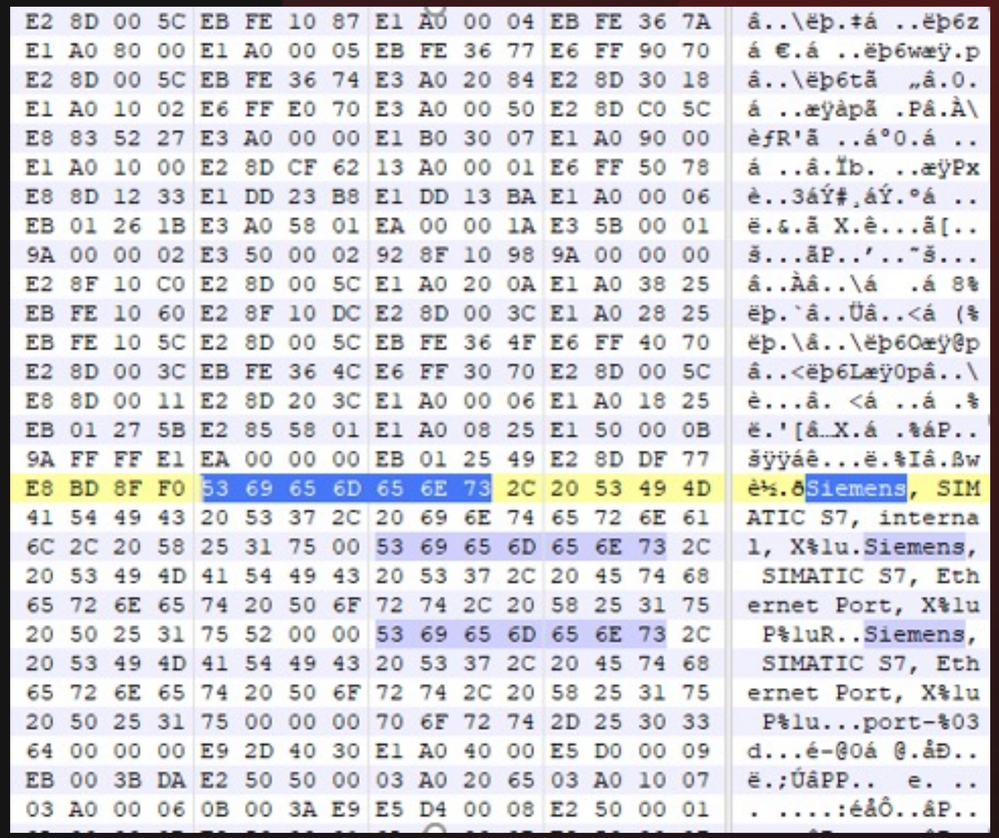
Dump size



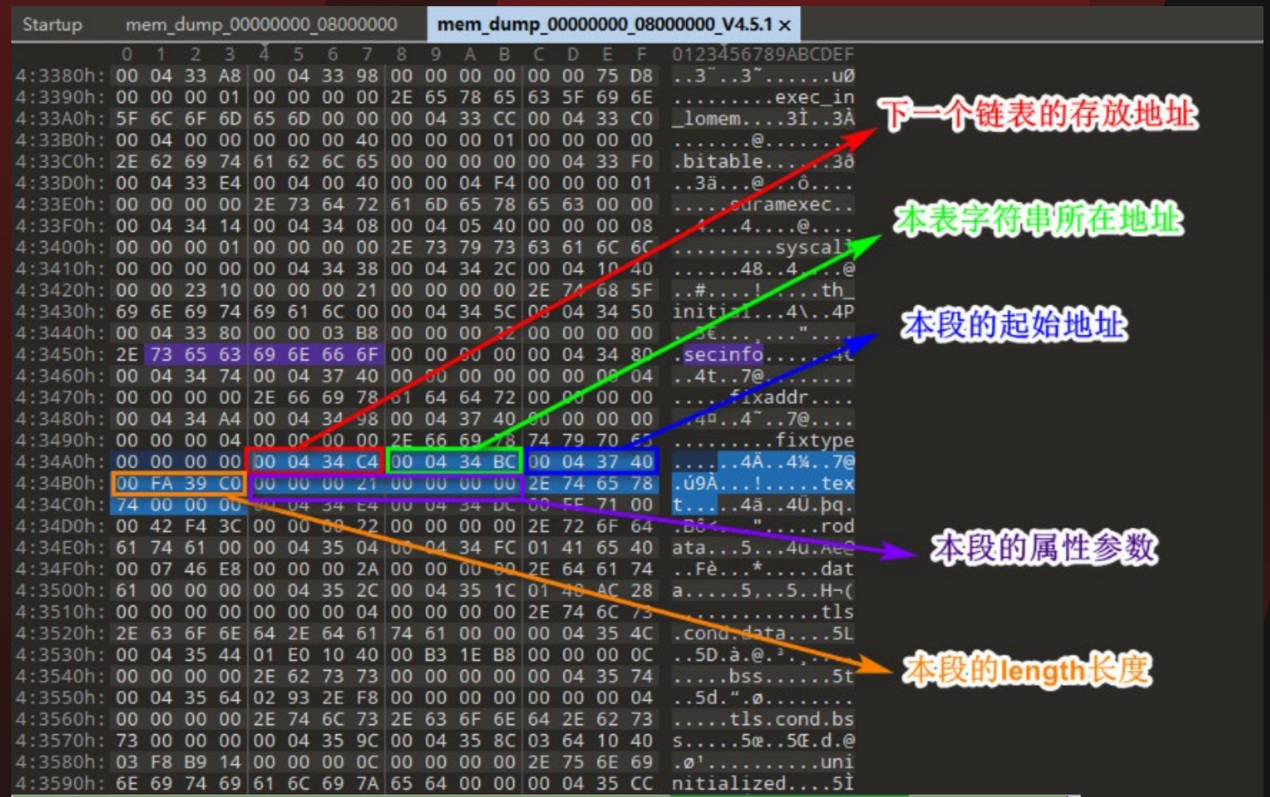
KCon

+1

利用设备漏洞提取固件-西门子S7-1200



- Dump的内存大小为128M字节；
- 内存中包含了起始的BootLoader代码128K；
- Secinfo的信息在固件文件中以链表形式存储，如右图所示；
- 内存中包含了固件代码、用户程序、访问保护密码hash信息、私钥信息等；



Module=1215C DC/DC/DC FW Version=V4.5.1			
段信息	起始地址	长度	
.text	0x00043740	0x00FA39C0	代码段
.rodata	0x00FE7100	0x0042F43C	只读数据
.data	0x01416540	0x000746E8	数据段
.tls.cond.data	0x0148AC28	0x00	
.bss	0x01E01040	0x00B31EB8	
.tls.cond.bss	0x02932EF8	0x00	
.uninitialized	0x03641040	0x03F8B914	

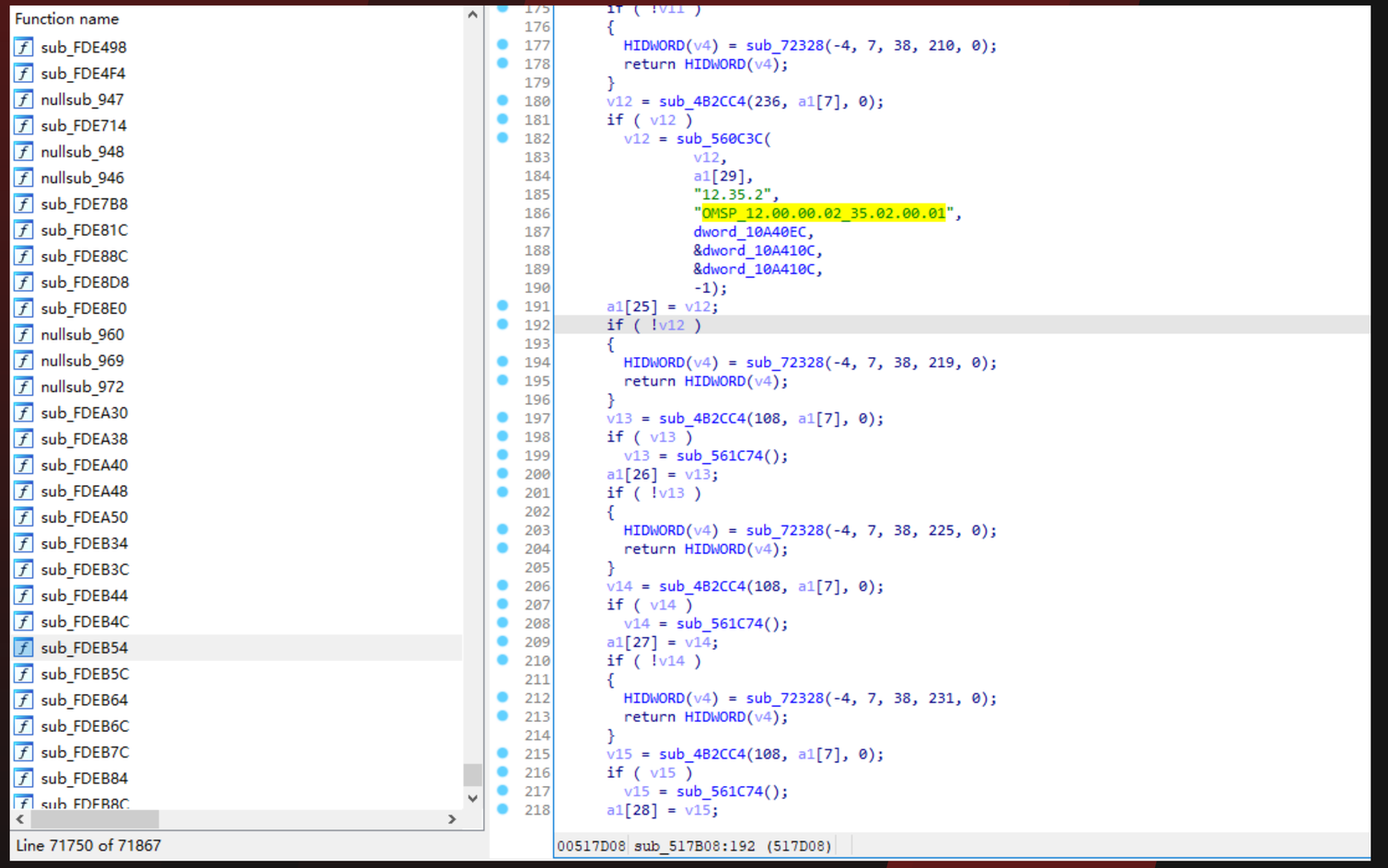
KCon

+1

利用设备漏洞提取固件-西门子S7-1200

固件文件初步分析方向

- 分析固件的处理算法LZP3,可以实现一套对所有S7-1200系列固件.upd格式的解压缩程序；
- 寻找S7CommPlus协议的处理部分，利用符号执行技术可以进行漏洞挖掘；
- 分析密码hash值，可以利用pass the hash绕过访问保护密码；
- 分析Miniweb协议解析代码部分，可以结合多种手段进行fuzzing测试；
- 分析并寻找所依赖的基础组件，比如基础协议栈、加密库等，从中间件入手进行脆弱性分析；
- ……

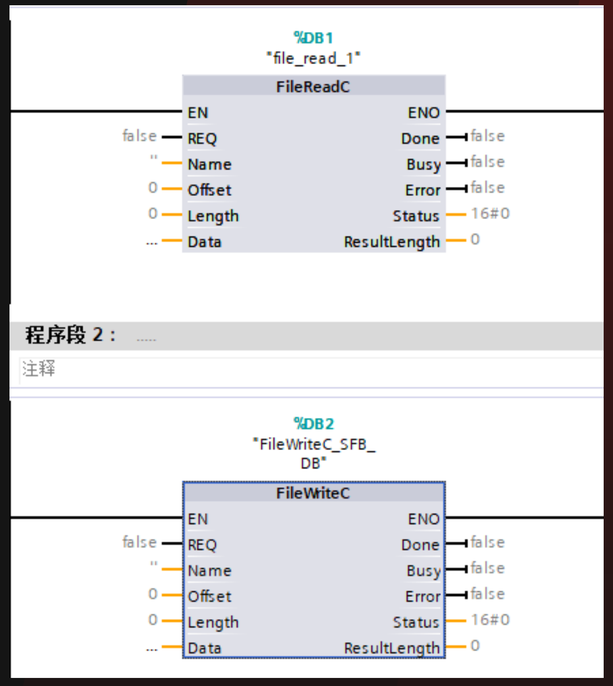


KCon

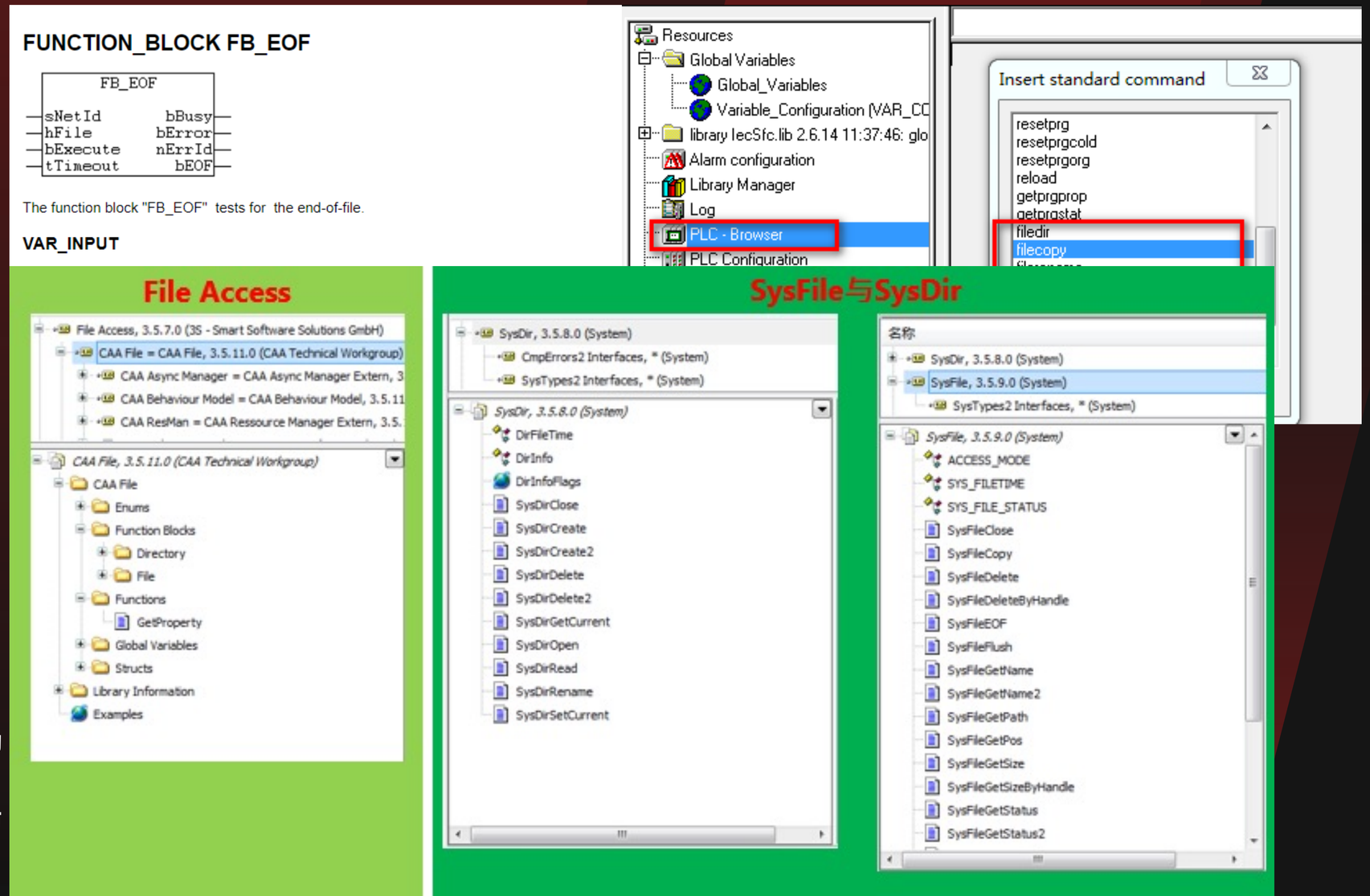
利用文件功能提取固件

+1

PLC有哪些文件功能？？？



- PLC中的文件操作一般对文本文件操作，最常见的为.txt/.csv/.log
- 应用场景为生产异常数据记录、效能数据记录、生产配方数据存储等



KCon

+1

利用文件功能提取固件

如何巧用文件功能提取固件？？？

以ABB厂商的AC500系列PLC为例讲解

分析PLC文件功能

- ✓ 搭建测试环境，初步了解研究目标具备的文件功能；
- ✓ 分析AC500文件功能与Codesys文件功能的差异；
- ✓ 熟悉各类文件指令的使用方法与范围；
- ✓ 重点研究拷贝指令的使用；

寻找目标所在位置

- ✓ 先厘清PLC中所有的disk设备；
- ✓ 各类disk中包含的内容；
- ✓ 找到目标文件所在的disk设备；

文件拷贝测试验证

- ✓ 是否支持文件list功能需要验证；
- ✓ 如若不支持文件list功能如何寻找目标文件；
- ✓ 找到一种办法可以验证disk下是否存在该文件的有效方法；
- ✓ 目标文件是否可以跨区域拷贝；

PLC读取文件至PC机

- ✓ 目标文件经过拷贝后所在的位置是否可以访问；
- ✓ 用什么方法将目标文件从PLC中拷贝到PC中？

固件正确性验证

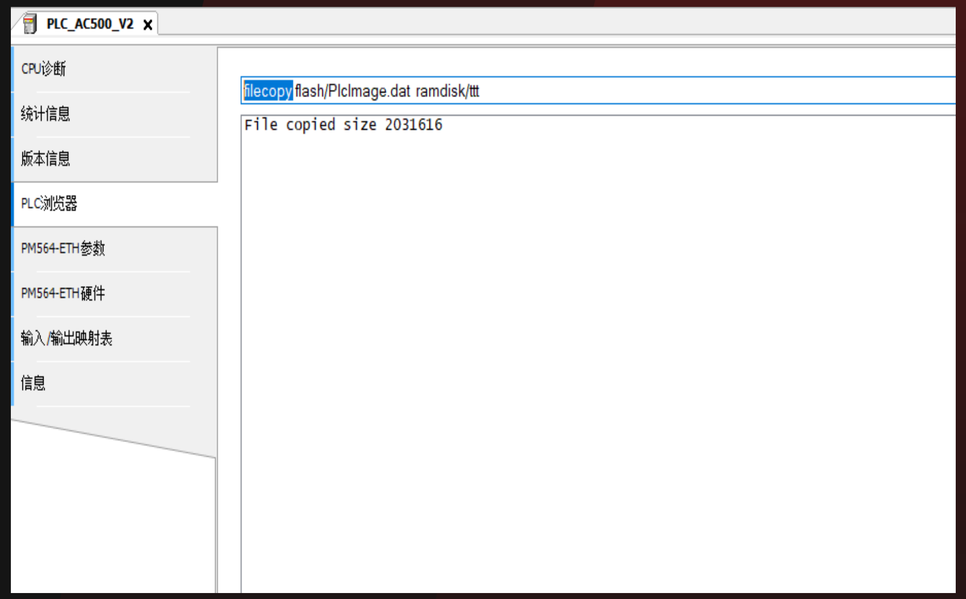
- ✓ 提取出的目标是否准确；
- ✓ 如何快速验证提取出的目标文件是预期文件；

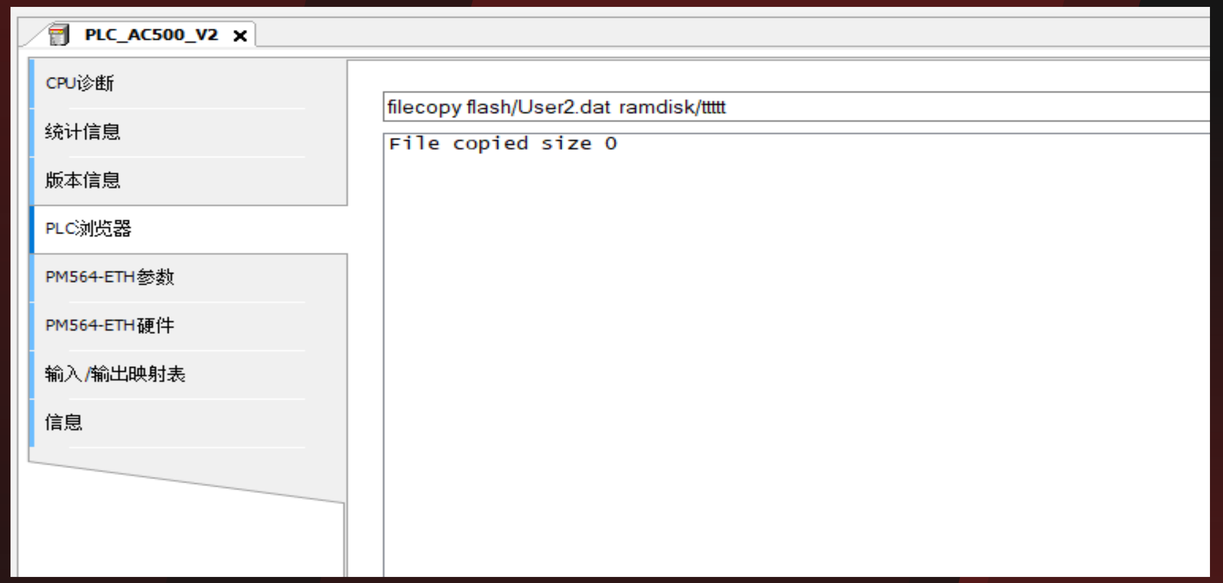
KCon

利用文件功能提取固件

利用filecopy指令验证文件存在性

一举两得，同时验证文件存在性和拷贝指令的使用



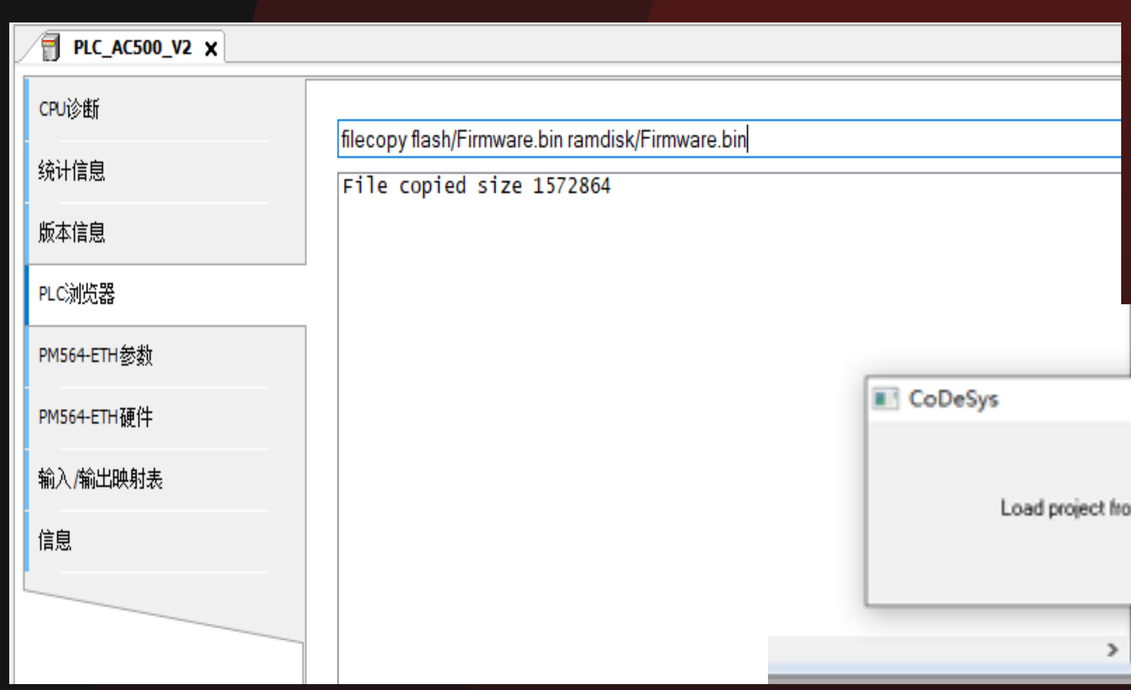


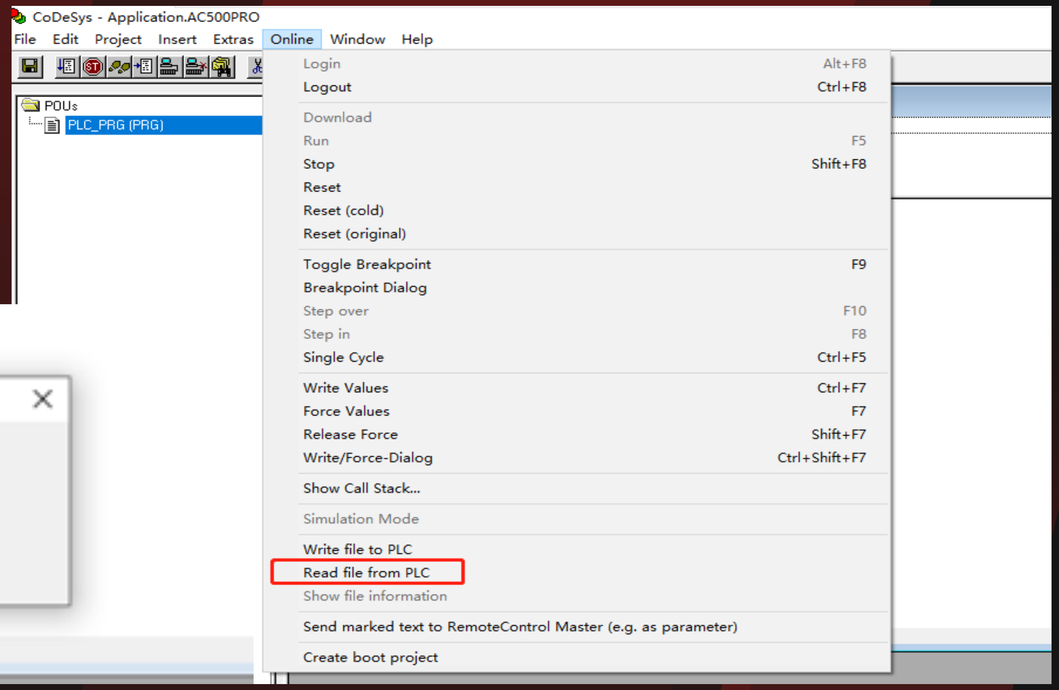
- Filecopy指令用于将目标文件从一个区域拷贝至另一个区域；
- 当猜测的文件存在于flash中时，拷贝过程完成后会显示拷贝文件的总大小（例如PlcImage.dat文件）

- 当猜测的文件未在flash中时，拷贝过程完成后会显示文件大小为0；
- 因此利用这种方法可以不停尝试猜测目标文件的文件名和拷贝过程；

KCon

利用文件功能提取固件





- 利用上述的方法，猜测固件名为firmware，文件格式为最常用的.bin，发现失败；最后是因为需要区分大小写，因此使用如上以上指令

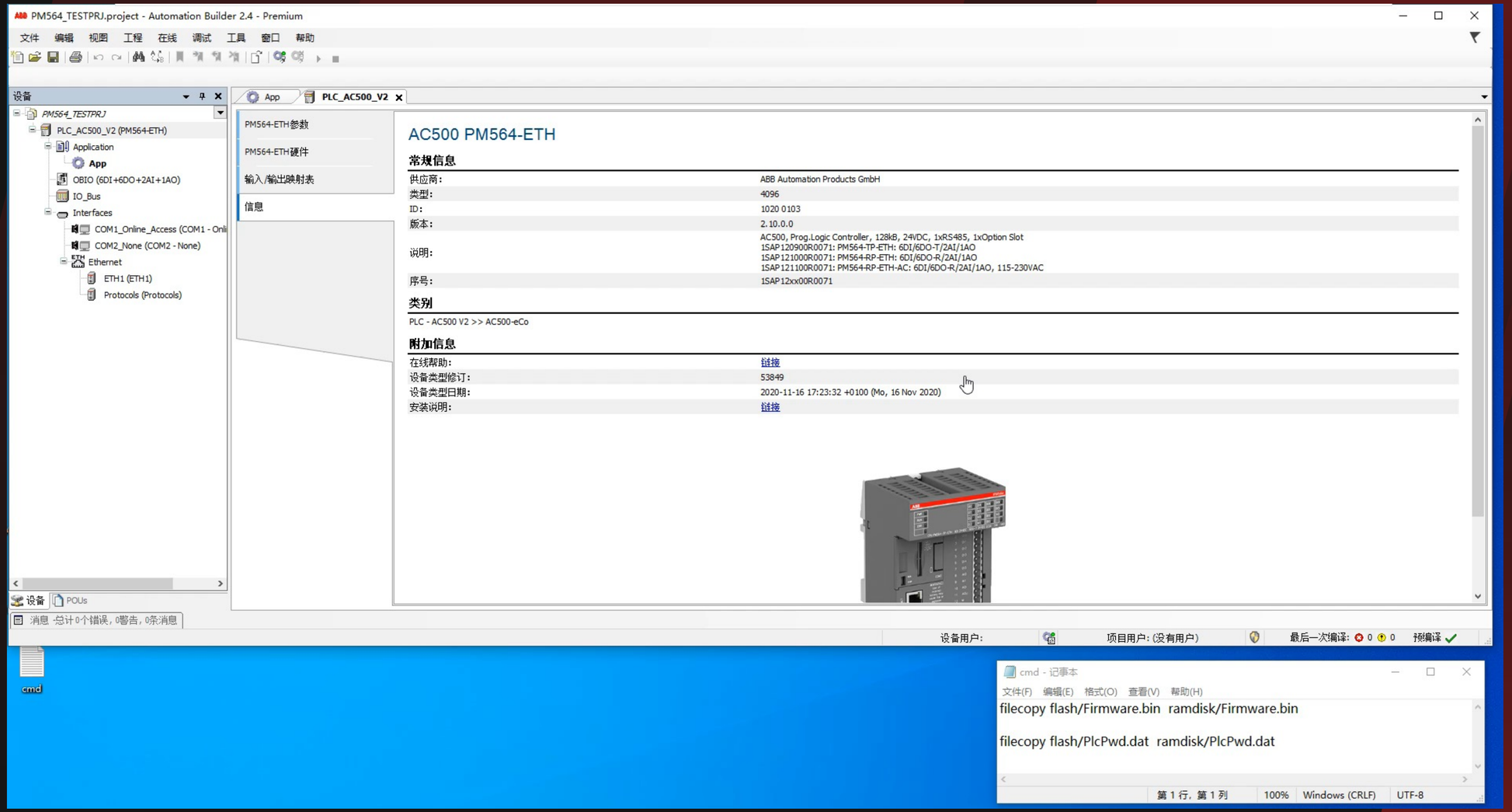
  filecopy flash/Firmware.bin ramdisk/Firmware.bin

- 目标文件从flash到ramdisk后，发现Codesys V2 IDE中附带有从PLC中读取文件的操作；
- 使用该方法，从PLC下的ramdisk中将Firmware拷贝至目标PC机桌面；

KCon

+1

利用文件功能提取固件



KCon

+1

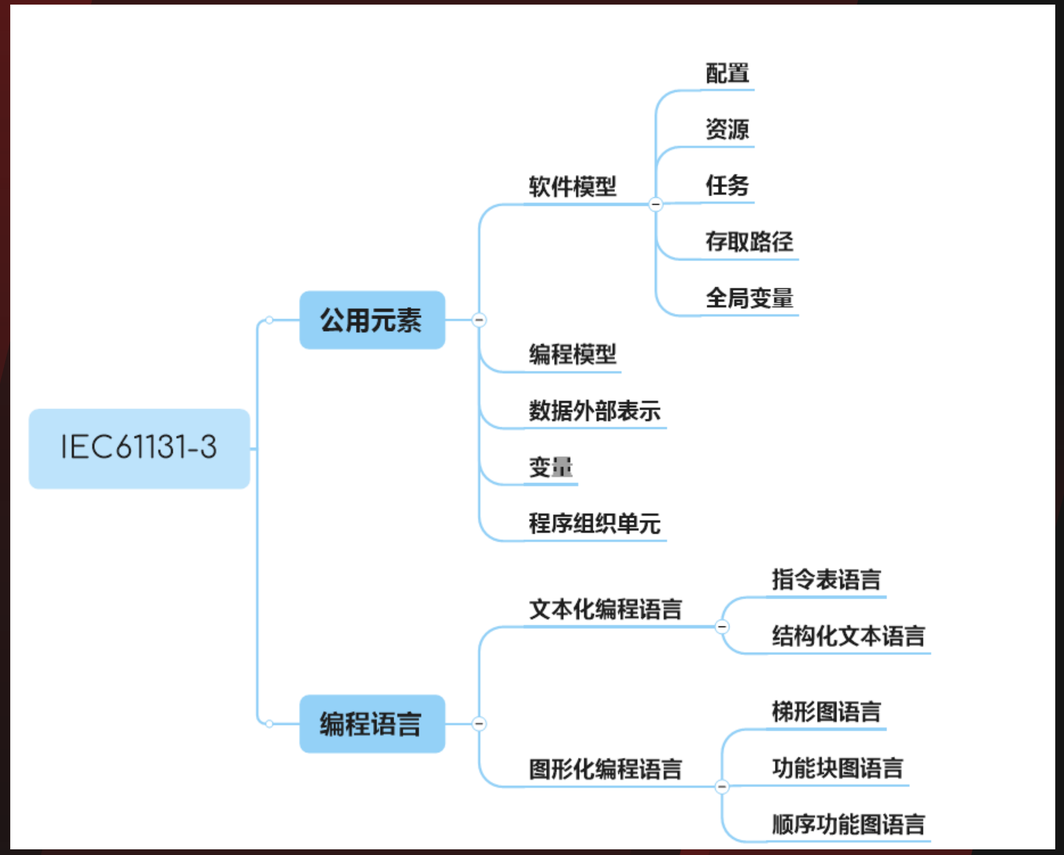
利用组态编程语言提取固件

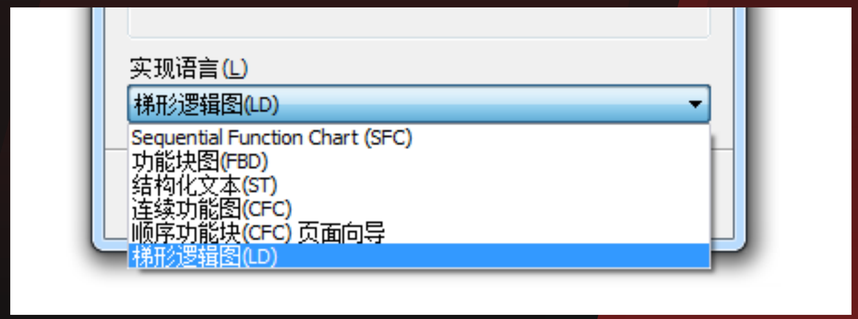
PLC的实现方式及编程语言分类

解释型PLC：编程软件会根据用户的编程语言生成中间文件，PLC采用解析语句表来完成指令的执行，下位机PLC侧有个执行指令的解释器，类似于虚拟机。典型的代表有西门子S7系列PLC，三菱FX系列PLC；

编译型PLC：编程软件会根据用户的编程语言生成PLC侧直接可以执行的二进制本地机器代码。典型的代表有施耐德PLC、罗克韦尔PLC；

区别	编译型	解释型
执行效率	高	低
开发难度	大	小
跨平台运行	难	易
反编译源文件	难	一般
无扰下装	难	易
防止克隆	好	差





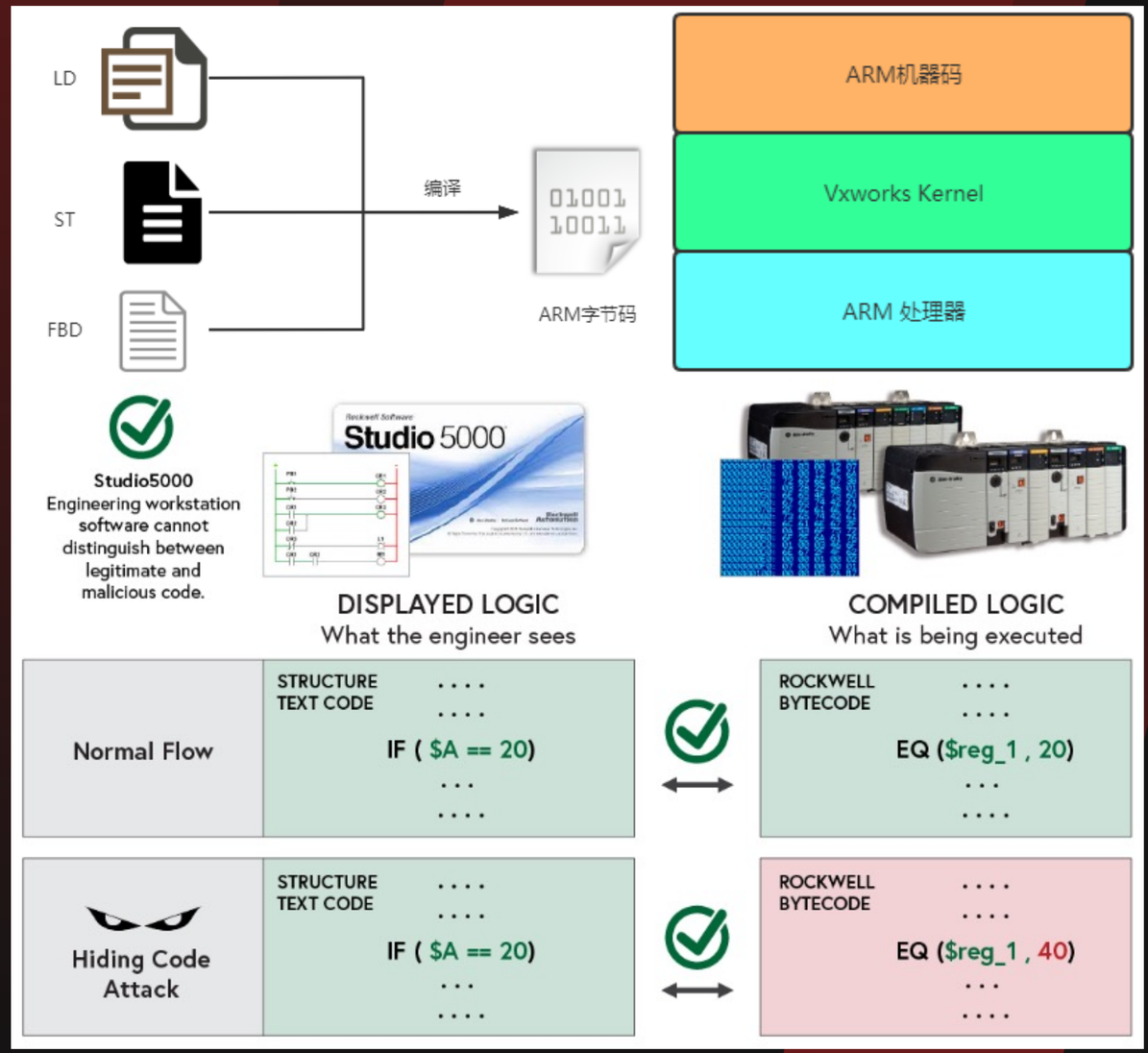
KCon

+1

利用组态编程语言提取固件

编译型PLC本地机器码引入的风险

- 不论是IEC-61131-3标准中的何种编程语言，在编译型PLC的IDE中均会被编译成机器码，并通过通信协议下装至PLC的特定区域执行；
- 基于ARM类型的编译型PLC，Airbus和claroty分别针对M340和Contrologix系列PLC开发出了类似于震网的代码执行攻击示例；
- 思路：替换恶意dll——设计恶意代码payload——下装payload至PLC侧——运行PLC；
- 恶意代码功能：根据需求可以有很多功能，比如添加后门账号、添加socks代理、定期回传特定区域数据等等；
- 本质的问题：**编译后的本地机器码可以不受限制的在处理器中执行；**

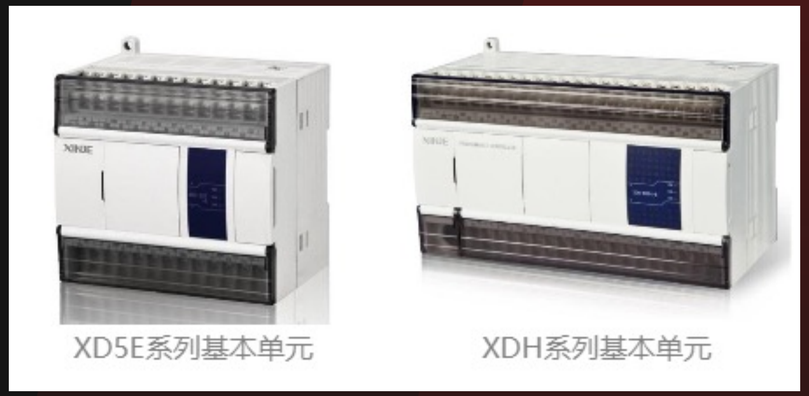


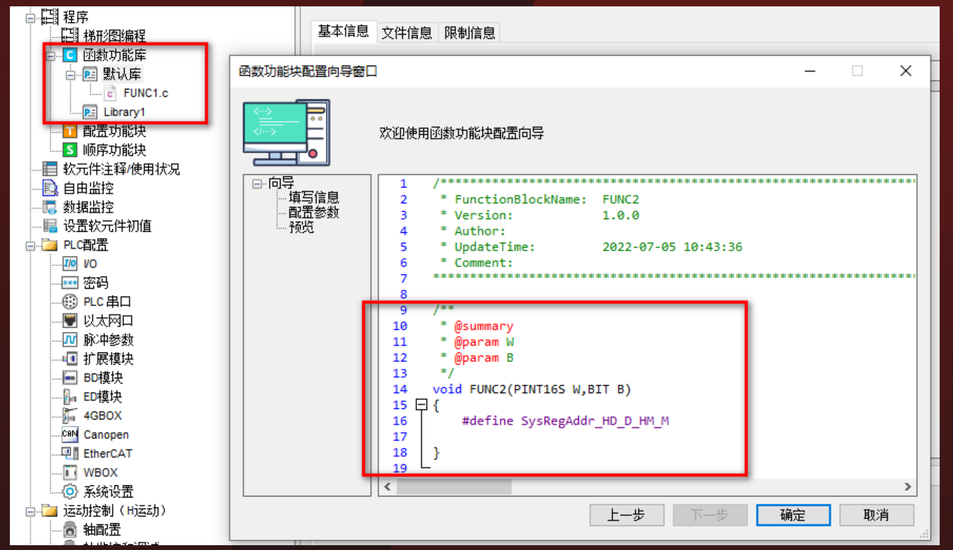
KCon

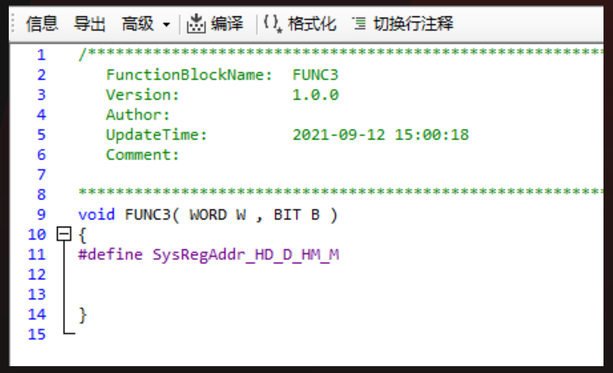
+1

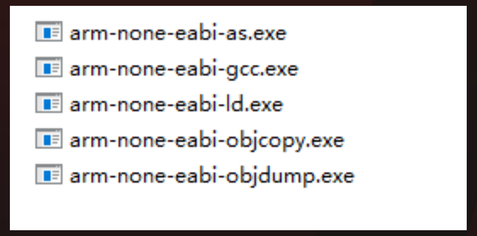
利用组态编程语言提取固件

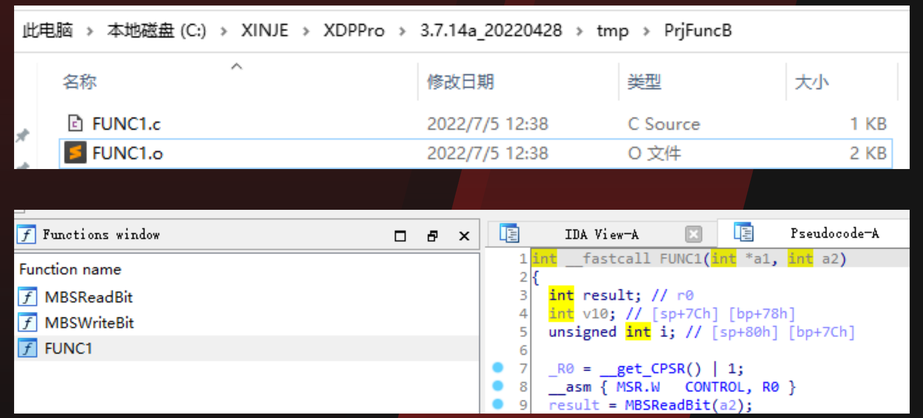
巧用高级语言访问内存地址内容









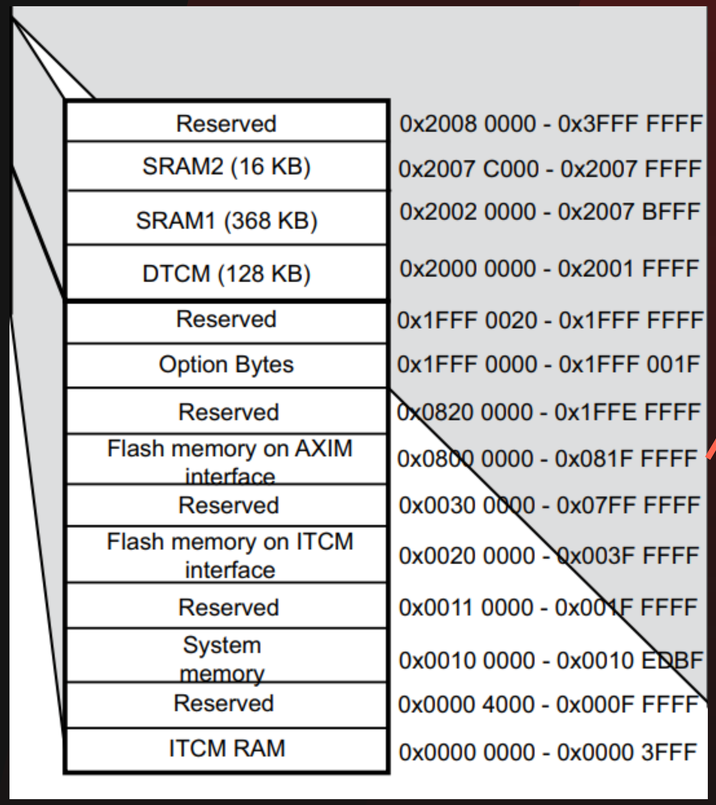


- 选取信捷XD5E系列PLC模块，支持C语言编程，可方便现场用户编写各类算法；
- C语言程序可作为库函数使用，库函数经过arm-gcc编译后为可执行的机器码文件；

KCon

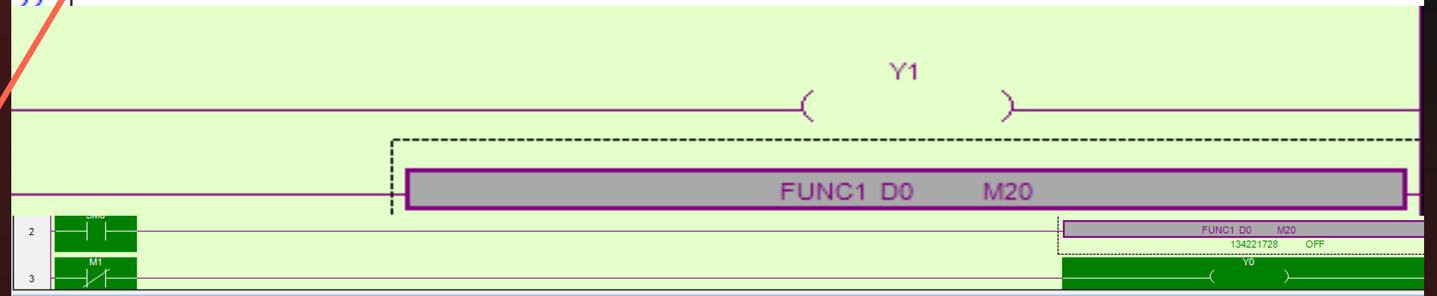
利用组态编程语言提取固件

如何确定基地址？读取size？



STM32F76xxx系列CPU芯片memory map

```
14   void FUNC1 ( PINT32U W, BIT B )
15   {                                          addr = ( unsigned int * ) W[0];
16       char buf[100];
17       unsigned int  *addr = NULL;            for ( int i = 0; i < size; i++ )
18       unsigned int base_addr = 0x08000000;   {
19       unsigned int size = 500;
20                                                  unsigned int val = addr[i];
21                                                  W[i + 5] = val;
```



PLC1-数据监控

监控 | 搜索: D0 | X | Y | M | S | SM | T | ET | C | HM | HS | HT | HC | HSC | D | SD | ID | QD | HD | HSD | FD | SFD | FS | SEM

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
D0	4000	2048	0	0	2000	2048	0	0	0	0
D10	-3197	16601	23633	16584	-17150	-19072	-2049	-25	-1825	4840
D20	26633	-1825	8956	-3055	16736	-2977	9184	-1357	-3197	16601
D30	23633	16584	-17150	-19184	4	11264	-12031	8193	-8146	-1900
D40	59	10240	-12027	8192	-1916	58	32	18112	18112	8194
D50	-1916	59	26656	26624	2112	64	26657	24584	32	-4096
D60	-1935	10241	-12031	8193	-8172	26656	26688	-3024	16528	26657
D70	24648	26656	26752	-4048	42	26657	24712	26656	26624	-4016
D80	1	26657	24584	32	-4096	-1716	-17136	-19184	4	11264

- 从PLC使用的CPU芯片的datasheet入手查看片内flash的起始地址；经过测试选择AXIM接口的1M字节flash
- 由于封装的读内存函数映射片区为D区，且考虑到读取内容过多容易引起错乱，因此每次读取2000字节内容；
- 将从flash中读取出的内容方式在了D10开始的寄存器片区中，存放大小为2000字节；

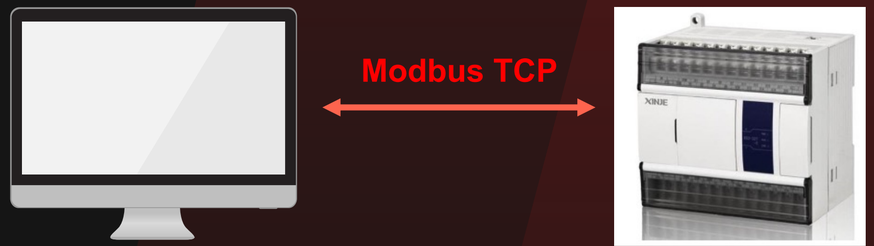
KCon

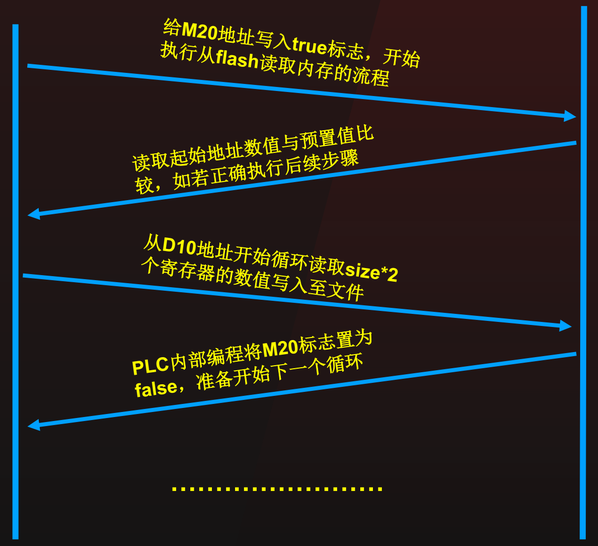
+1

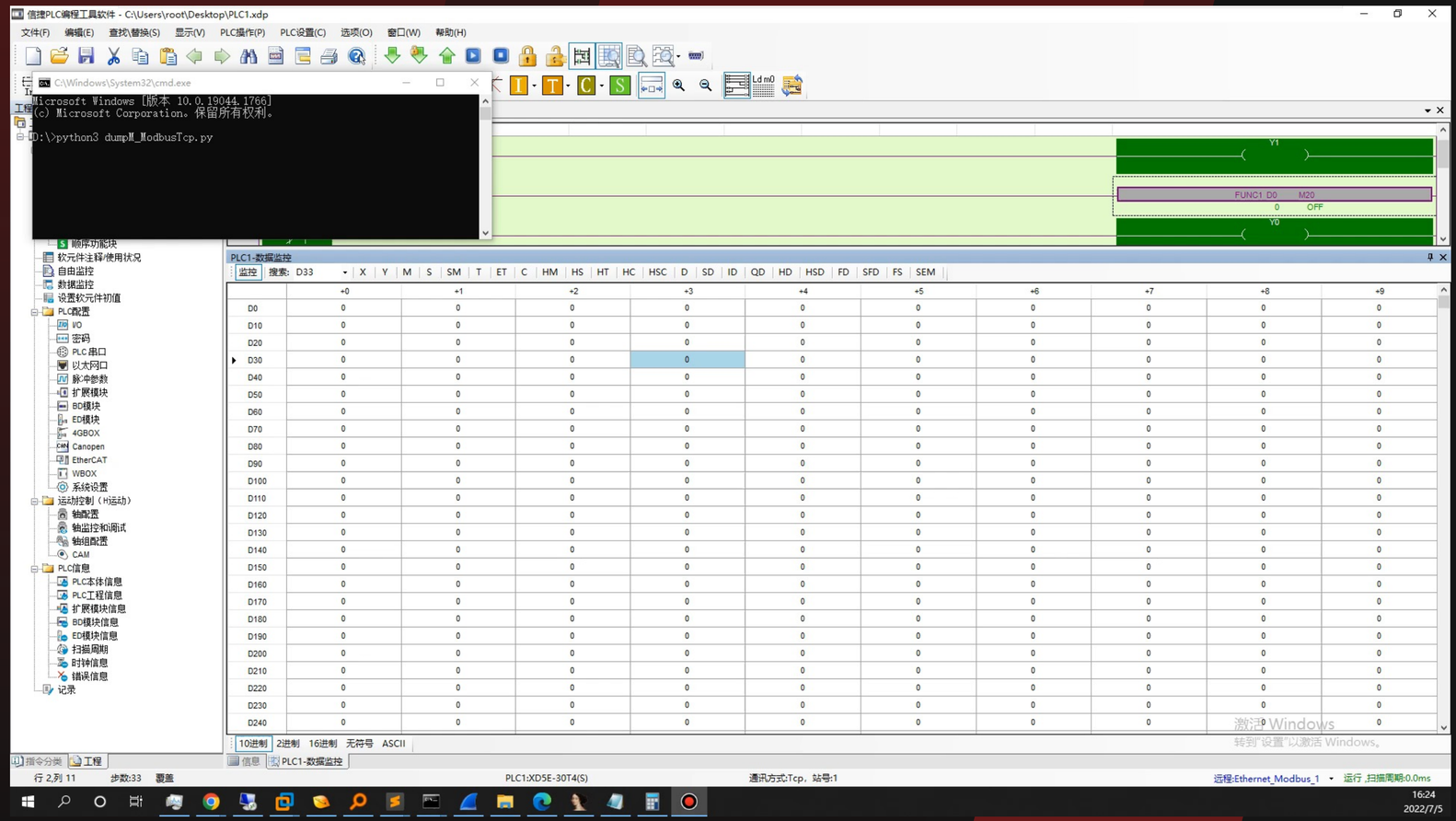
利用组态编程语言提取固件

如何dump固件至文件中？

每次只能读取2000字节，而且只能读取从D10开始的区域，如何自动化的获取flash内容到可读文件中？







KCon

+1

固件提取方法总结

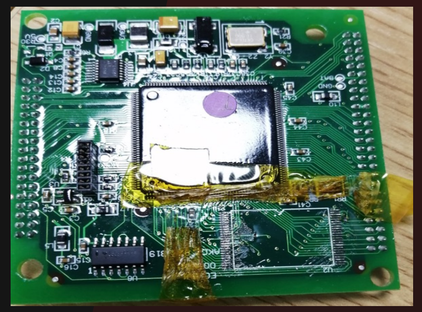
KCon

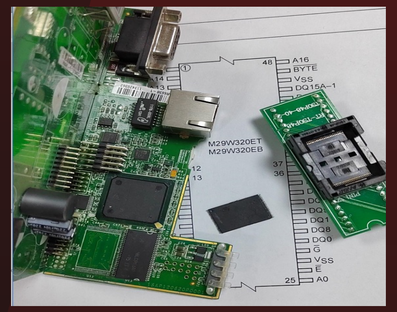
+1

➢ Flash内容读取法

适用于价值在承受范围内的工控类设备，分析硬件组成选择适合的读取方法：引脚法、拆焊法（flash芯片&CPU芯片）、JLINK读取等等；

实战验证：ControlLogix PLC，安控RTU，ABB AC500系列PLC

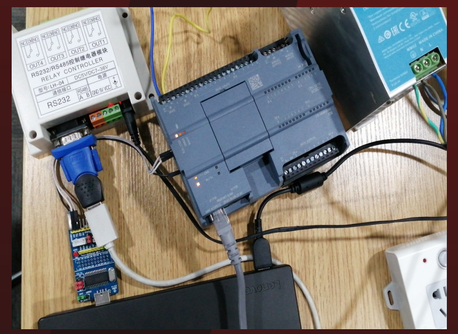




➢ 利用设备漏洞dump内存获得固件；

适用存在历史漏洞，该漏洞是读写内存或远程代码执行类型，通过对漏洞加以利用就可以完整dump出整个内存；

实战验证：M580,S7-1200,S7-1500（正在研究中……）



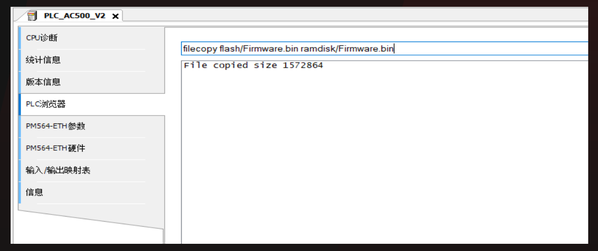
```
int __fastcall bug_A9758(int a1)
{
  int v1; // r4
  int v2; // r4
  int v3; // r0
  int v4; // r0
  int v5; // r1
  bool v6; // cc
  int v7; // r0
  int v8; // r1
  bool v9; // zf
  int result; // r0
  int v11; // r4
  unsigned int *v12; // r1
  unsigned int v13; // r2
  char v14[8]; // [sp+4h] [bp-20h] BYREF
  unsigned int *v15; // [sp+Ch] [bp-18h]
  int v16; // [sp+14h] [bp-10h]

  v1 = a1 + 16;
  sub_7F4(a1, v14);
  if ( (unsigned int)sub_A9E1C(v1, (unsigned __int8)v14[6], 448, 1, 0) < 0x1C0 )
    return sub_A2620(9506, *(_DWORD *)(dword_1084924C + 16), 0);
  v2 = v16;
  v3 = sub_A9520(v16);
  v4 = sub_AB128(v3, *(unsigned __int16 *)(v2 + 4));
  v5 = *(_DWORD *)(v2 + 12);
  v6 = v5 <= 0;
  if ( v5 >= 0 )
    v6 = 0 * v4 - v5 <= 0;
  if ( v6 )
    return memset_47DF0(v15, 0, 52);
```

CVE-2020-15782

➢ 利用文件功能获取固件

适用于存在文件拷贝与读取功能的设备，寻找设计中的缺陷，将文件拷贝至不受保护的区域后进行读取即可获得固件

实战验证：ABB AC500系列PLC



➢ 利用组态编程功能获取固件

适用于支持高级语言编译型的工控设备，通过编程将内存地址的内容读取到设备的合法区域中，再利用设备支持的协议将内存内容读取到文件中；

实战验证：信捷XD5E系列PLC

条条大路通罗马，没有最好的只有最适合的方法！选用最快捷、损失最小、最拿手的方法突破工控设备研究的第一步！！！

KCon

+1

感谢您的观看

THANK YOU FOR YOUR WATCHING

KCon 2022 黑客大会