

The spear to break the security wall of S7CommPlus

Cheng Lei, Li Donghong, Ma Liang
NS-Focus

Abstract. Siemens PLCs was widely used in industrial control system(ICS). The new version of Siemens PLCs like S7-1500 and S7-1200v4.0 used an encrypted protocol names S7CommPlus to prevent replay attacks. In this paper, based on reverse debugging techniques, we will demonstrate the encryption algorithms of S7CommPlus and program a MFC to control the Siemens PLC. Finally, some more security protective measures have been proposed according to our research.

1. Introduction.

Industrial Control System involves national level critical infrastructure and requires highly Security. In the past few years, attacks against industrial control systems (ICS) have increased year over year. Stuxnet in 2010 exploited the insecurity of the S7Comm protocol, the communication protocol used between Siemens Simatic S7 PLCs to cause serious damage in nuclear power facilities. After the exposure of Stuxnet, Siemens has implemented some security reinforcements into the S7Comm protocol. The current S7CommPlus protocol implementing encryption has been used in S7-1200 V4.0 and above, as well as S7-1500, to prevent attackers from controlling and damaging the PLC devices.

Is the current S7CommPlus a real high security protocol? This talk will demonstrate a spear that can break the security wall of the S7CommPlus protocol. First, we use software like Wireshark to analyze the communications between the Siemens TIA Portal and PLC devices. Then, using reverse debugging software like WinDbg and IDA we can break the encryption in the S7CommPlus protocol. Finally, we write a MFC program which can control the start and the stop of the PLC, as well as value changes of PLC's digital and analog inputs & outputs. This paper is based on the Siemens SIMATIC S7-1200v4.1.

2. Related Work

At Black Hat USA 2011, Dillon Beresford demonstrated how to use

reconnaissance, fingerprinting, replay attacks, authentication bypass methods, and remote exploitation to attack a Siemens Simatic S7-300 PLCs. These PLCs use S7Comm protocol which does not contain any security protection. At Black Hat USA 2015, Ralf Spenneberg et. al. demonstrated a worm lives and runs on the Simatic S7-1200v3 PLCs. These PLCs use the early S7CommPlus protocol with a simple mechanism to prevent replay attacks.

3. Siemens PLCs

Siemens PLCs are widely used in industrial control systems, like power plants, fuel gas station, water and waste.

3.1 Programmable Logic Controllers

Programmable Logic Controllers (PLC) is responsible for process control in industrial control system. A PLC contains a Central Processing Unit (CPU), some digital/analog inputs and outputs modules, communication module and some process modules like PID. Engineers programed user programs for automated process control in PLC software and then downloaded the user program to the PLC. The authorized engineers can also run or stop the PLCs from PLC software.

3.2 Siemens PLCs protocols

Siemens PLCs use a private protocol to communicate. It is a binary protocol utilizing both TP/RT and ISO8073. Typically, both of these protocols use port 102/TCP.

The newest version of Wireshark(V2.1.1) supports Siemens PLC protocols recording that will permit the analysis of message frames. Siemens PLC protocol has 3 versions, S7Comm protocol, early S7CommPlus protocol and new S7CommPlus protocol. S7Comm protocol is used in the communication among S7-200, S7-300 and S7-400 PLCs. This protocol did not involve any anti-replay attack mechanism and can be easily exploit by attackers. The early S7CommPlus protocol used in the communication among S7-1200v3.0 is more complicated than S7Comm protocol and use two-byte field called session ID for anti-replay attack. However, the session ID is too easy to calculate. The new S7CommPlus protocol used in the communication among S7-1200v4.0 and S7-1500 has a complex encryption part to against replay attack. In this paper, we will focus on the encryption part of S7CommPlus.

3.3 TIA Portal

TIA Portal is the configuration and programming software for Siemens PLCs. Engineers rely on this software to design logic and program to control the process attached to the PLC. The software offers the programmer the ability to configure hardware parameters, such as Profinet parameters, communication type, diagnostics. Authorized engineers can also run or stop the PLCs, monitor and modify the input/output values.

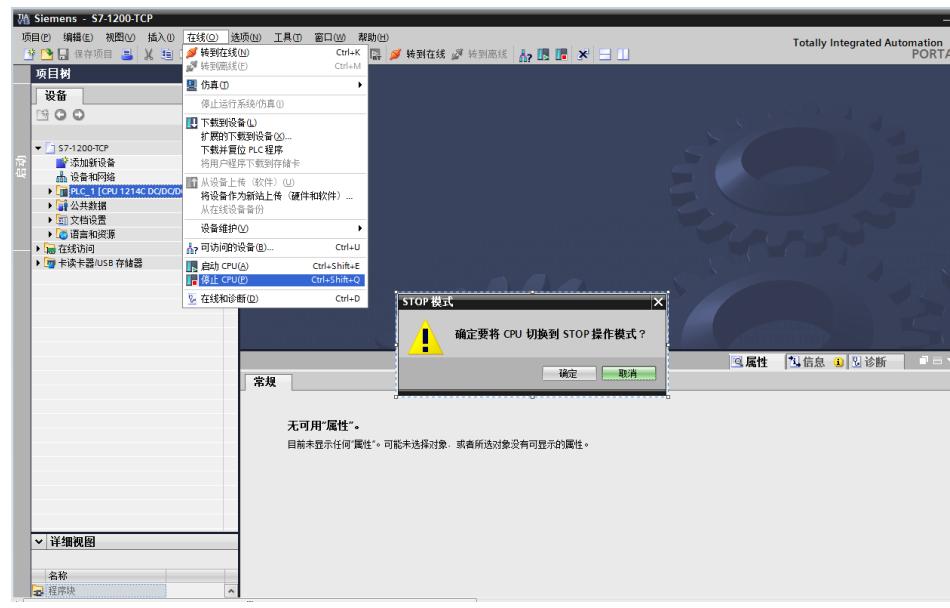


Figure 3.1 TIA Portal CPU STOP

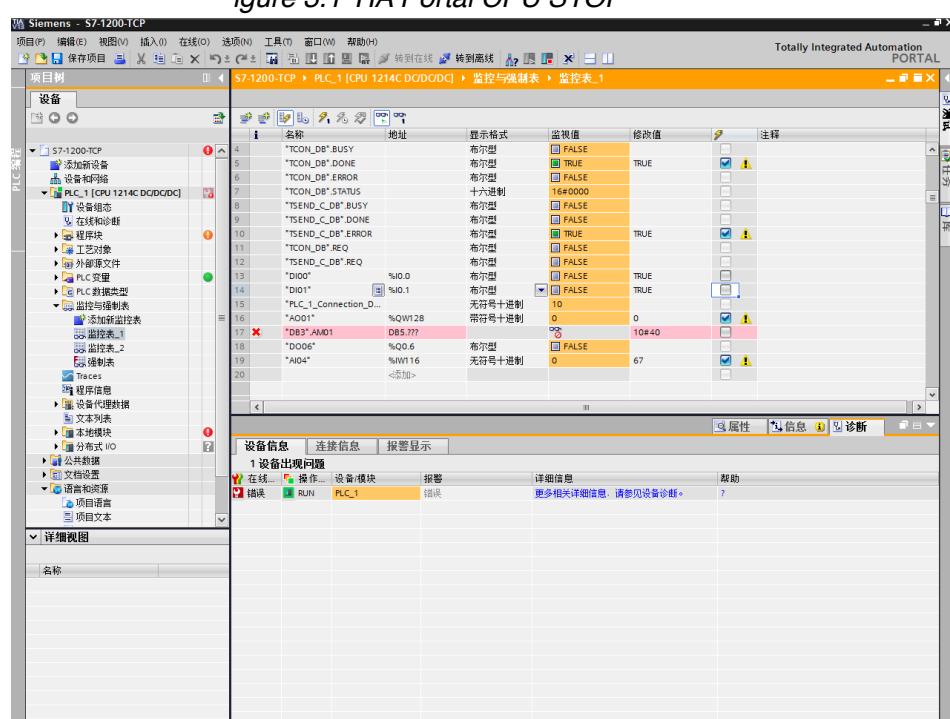


Figure 3.2 TIA Portal value monitor and modify

4. Replay Attacks

Replay attacks have been widely used in PLC attacks. We build up a small net environment with a TIA Portal PC, a PLC and a hub. First, click the Stop PLC button in TIA Portal to stop the PLC. Then launch the Wireshark or other packet capturing tool to capture the packets between PC and PLC. Once the PLC has stopped, stop capturing the packets. Use the packets we have already obtained and send these packets back to any PLC in sequence, the PLC could be controlled with these packets.

It is also possible for attackers to run PLCs, monitor or modify the analog/digital input/output values, download user program or system program, monitor the diagnostics of PLC.

No.	Time	Source	Destination	Protocol	Length	Info
1049	2017-02-24 13:37:26.264282	10.65.96.89	10.65.60.73	TCP	66	5208+102 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
	TCP Connection	10.65.60.73	10.65.96.89	TCP	60	102+5208 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460
1022	2017-02-24 13:37:26.265899	10.65.96.89	10.65.60.73	TCP	54	5208+102 [ACK] Seq=1 Ack=1 Win=64240 Len=0
1033	2017-02-24 13:37:26.267364	10.65.96.89	10.65.60.73	COTP	89	CR TPDU src-ref: 0x0003 dst-ref: 0x0000
	COTP Connection	10.65.60.73	10.65.96.89	COTP	89	CC TPDU src-ref: 0x0001 dst-ref: 0x0003
1026	2017-02-24 13:37:26.276317	10.65.96.89	10.65.60.73	S7COMM-PLUS	289	4+5208 PDU-Type: [Connect] Op: [Request] Function: [CreateObject] Se...
1027	2017-02-24 13:37:26.286598	10.65.60.73	10.65.96.89	S7COMM-PLUS	251	4+5208 PDU-Type: [Connect] Op: [Response] Function: [CreateObject] S...
1028	2017-02-24 13:37:26.287630	10.65.96.89	10.65.60.73	COTP	61	61 DT TPDU (0) [COTP fragment, 0 bytes]
1029	2017-02-24 13:37:26.331976	10.65.96.89	10.65.60.73	S7COMM-PLUS	472	4+5208 PDU-Type: [Data] Op: [Request] Function: [SetMultiVariables] ...
1030	2017-02-24 13:37:26.360397	10.65.60.73	10.65.96.89	TCP	60	102+5208 [ACK] Seq=233 Ack=694 Win=8192 Len=0
1054	2017-02-24 13:37:26.459946	10.65.60.73	10.65.96.89	S7COMM-PLUS	86	4+5208 PDU-Type: [Data] Op: [Response] Function: [SetMultiVariables]...
1056	2017-02-24 13:37:26.460261	10.65.96.89	10.65.60.73	COTP	61	61 DT TPDU (0) [COTP fragment, 0 bytes]
1072	2017-02-24 13:37:26.556614	10.65.60.73	10.65.96.89	TCP	60	102+5208 [ACK] Seq=265 Ack=704 Win=8192 Len=0
1092	2017-02-24 13:37:26.693001	10.65.96.89	10.65.60.73	S7COMM-PLUS	155	4+5208 PDU-Type: [DataFW1_5] Op: [Request] Function: [GetVarSubStrea...
1093	2017-02-24 13:37:26.697851	10.65.60.73	10.65.96.89	S7COMM-PLUS	129	4+5208 PDU-Type: [DataFW1_5] Op: [Response] Function: [GetVarSubStrea...
1094	2017-02-24 13:37:26.697987	10.65.96.89	10.65.60.73	COTP	61	61 DT TPDU (0) [COTP fragment, 0 bytes]
1150	2017-02-24 13:37:27.081996	10.65.96.89	10.65.60.73	S7COMM-PLUS	155	4+5208 PDU-Type: [DataFW1_5] Op: [Request] Function: [SetVariable] S...
1151	2017-02-24 13:37:27.087581	10.65.60.73	10.65.96.89	S7COMM-PLUS	118	4+5208 PDU-Type: [DataFW1_5] Op: [Response] Function: [SetVariable] ...
	S7CommPlus Function	27.087691	10.65.96.89	COTP	61	61 DT TPDU (0) [COTP fragment, 0 bytes]
	:-Stop PLC	27.157371	10.65.60.73	TCP	60	102+5208 [ACK] Seq=1221 Ack=1780 Win=8192 Len=0
1163	2017-02-24 13:37:27.246673	10.65.96.89	10.65.60.73	S7COMM-PLUS	149	4+5208 PDU-Type: [DataFW1_5] Op: [Request] Function: [DeleteObject] ...
1165	2017-02-24 13:37:27.251266	10.65.60.73	10.65.96.89	S7COMM-PLUS	121	4+5208 PDU-Type: [DataFW1_5] Op: [Response] Function: [DeleteObject]...

Figure 4.2 Stop PLC communication sequence

Figure4.1 shows the communication sequence packets when stopping the PLC using Wireshark. We separated these packets into 4 parts, TCP Connection packets, COTP Connection packets, S7CommPlus Connection packets and S7CommPlus Function packets. Performance as TIA Portal, first establish the TCP connection and COTP connection to the target PLC. Then, send the two S7CommPlus connection packets. After the S7CommPlus connection was established, the S7CommPlus function packets could be used to control the target PLC, or read/write the PLC's input/output values.

5. S7CommPlus Protocol

Siemens S7-1200v4.0 and S7-1500 use the new S7CommPlus protocol including the S7CommPlus Connection packets and S7CommPlus Function packets. Every packets used by S7CommPlus protocol has a similar structure.

	Type	Sub-Type	Sequence		PDU Type		Data Length
			Number	Protocol ID	72 01 00	a3 82	
0030	fa cd b2 29 00 00 03 00 00 eb 02 f0 80	72 01 00					...).....r..
0040	dc 31 00 00 04 ca 00 00 00 01	00 00 01	20 36 00				.1..... 6..
0050	00 01 1d 00 04 00 00 00 00 00	a1 00 00 00	d3 82			
0060	1f 00 00 a3 81 69 00 15 15 53 65 72 76 65 72 53					i... .ServerS
0070	65 73 73 69 6f 6e 5f 31 43 39 43 33 38 30	a3 82					ession_1 C9C380..
0080	21 00 15 35 31 3a 3a 3a 36 2e 30 3a 3a 49 6e 74						!.51::: 6.0:::Int
0090	65 6c 28 52 29 20 45 74 68 65 72 6e 65 74 20 43						el(R) Et hernet C
00a0	6f 6e 65 63 74 69 6f 6e 20 49 32 31 37 2d 4c						onnectio n I217-L
00b0	4d 2e 54 43 50 49 50 2e 31 a3 82 28 00 15 00 a3						M.TCPIP. 1..(....
00c0	82 29 00 15 00 a3 82 2a 00 15 13 43 48 45 4e 47						.).....* ...CHENG
00d0	4c 45 49 2d 50 43 5f 31 38 35 39 39 32 31 a3 82						LEI-PC_1 859921..
00e0	2b 00 04 01 a3 82 2c 00 12 01 c9 c3 80 a3 82 2d						+....., ..
00f0	00 15 00 a1 00 00 00 d3 81 7f 00 00 a3 81 69 00					i..
0100	15 15 53 75 62 73 63 72 69 70 74 69 6f 6e 43 6f						..Subscr iptionCo
0110	6e 74 61 69 6e 65 72 a2 a2 00 00 00 00 72 01 00						ntainer.r..
0120	00						.

Figure 5.1 First S7CommPlus Connection Request Packet

Figure 5.1 shows the first S7CommPlus Connection Packet. Byte 0x72 represents the start of the S7CommPlus packet. Then following the PDU Type byte, 0x01 means this packet is a connection packet. The Data Length field does not take into account the frame boundary. Following the Data Length is the type of this packet, 0x31 means this packet is a request packet. The Sub-type byte further specifies this packet. The sequence number is incremented for each message. Additional data is transferred in separate attribute blocks begin with the two bytes “0xa3, 0x8x”. Frame Boundary is used as the end of S7CommPlus packet.

	Type:Response	Object ID	
0030	20 00 70 5c 00 00 03 00 00 c5 02 f0 80 72 01 00		.p\.....r..
0040	b6 32 00 00 04 ca 00 00 00 01 36 00 02 87 0f 87		.2..... ..6.....
0050	1a a1 00 00 01 20 82 1f 00 00 a3 81 69 00 15 02	i...
0060	30 31 a3 82 2b 00 04 82 80 80 80 00 a3 82 2d 00		01..+.... ..
0070	15 10 4f 4d 53 50 2e 52 45 4c 2e 37 30 37 30 2e		..OMSP.R EL.7070.
0080	31 34 a3 82 2f 10 02 14 1c 16 84 ed 01 be 4f fc		14../.... ..0..
0090	2d dd 3c 34 d4 a1 83 aa 3b 61 56 03 a3 82 32 00		-..<4.... ;aV...2.
00a0	17 00 00 01 3a 82 2c 22 21 22 10 22 3c 00 04 83	;.. ..@.<...
00b0	00 82 3d 00 04 84	04 84 80 c1	..=..... @.>....
00c0	00 82 3f 00 15 1tValue Array	20 32 31 34	..?..1; 6E57 214
00d0	2d 31 41 47 34 30 2a 30 58 42 30 20 3b 56 34 2e		-1AG40-0 XB0 ;V4.
00e0	31 82 40 00 15 05 32 3b 38 31 38 82 41 00 03 00		1.@@...2; 818.A...
00f0	03 00 a2 00 00 00 00 72 01 00 00	r ...

Figure 5.2 First S7CommPlus Connection Response Packet

Figure 5.2 shows the first S7CommPlus Connection response packet. Type byte 0x32 means this packet is a response packet. The 17th and 18th bytes

presents the Object ID. The 17th byte is constant with the value of 0x87 and the 18th byte is a random byte ranges from 0x06 to 0x7f generated by the PLC. The 76th to 95th bytes presents the value array. This value array is a random array generated by the PLC.

Session ID											
0030	fa	08	b2	e0	00	00	03	00	01	a2	02
0040	93	31	00	00	05	42	00	00	00	02	00
0050	00	03	8f	02	02	8e	26	82	32	01	00
0060	8e	09	00	04	00	8e	0a	00	02	00	8e
0070	07	21	8e	22	00	05	de	d0	cd	b0	c8
0080	23	00	04	84	82	10	8e	24	00	04	00
0090	00	00	07	21	8e	22	00	05	c1	e5	ba
00a0	ec	8e	23	00	04	84	82	01	8e	24	00
00b0	00	14	00	81	34	ad	de	e1	fe	b4	00
00c0	00	01	00	00	00	ec	dc	10	49	10	d7
00d0	00	00	00	00	00	1a	73	08	1f	09	6b
00e0	00	00	00	00	00	01	99	ec	e4	62	a6
00f0	bf	fa	d9	85	44	bd	b0	11	80	6c	95
0100	60	55	35	97	3e	5a	f6	0c	fb	85	57
0110	d6	8b	1b	e1	First Connection Encryption						3e
0120	f9	53	59	73	ef	du	jl	ju	20	40	01
0130	cb	10	c4	f0	42	48	1b	f7	bc	d5	a7
0140	f7	ff	66	bf	3f	1d	4b	2d	52	b2	1a
0150	4c	85	20	bf	55	9c	2d	7e	c8	01	ce
0160	9d	e1	7a	6f	74	e9	95	66	82	00	02
0170	3a	82	3b	00	04	83	cc	cc	cc	cc	cc
0180	04	84	80	c1	00	82	cc	cc	cc	cc	cc
0190	15	00	82	40	00	15	1a	31	3b	36	45
01a0	34	2d	31	41	47	34	30	2d	30	58	42
01b0	30	82	41	00	03	00	03	00	30	3b	56
01c0	12	00	00	00	00	89	6a	00	13	00	89
01d0	00	00	00	00	72	02	00	00	6b	00	04
									00	04	00

Figure 5.3 Second S7CommPlus Connection Request Packet

Figure 5.3 shows the second S7CommPlus Connection packet. The 16th and 17th, 21th and 22th bytes is called Session ID. The 16th and 21th byte is constant with the value of 0x03. The 17th and 22th byte is calculated by TIA Portal with the following formula:

$$\text{Session ID} = \text{Object ID} + 0x80$$

In the second S7CommPlus Connection packet, there are two variable array, we called them Connection Encryption arrays. These two arrays are calculated by TIA Portal and we will talk this in the next chapter.

	Encryption length	Encryption Part	
0030	f6 6c b1 a3 00 00 03 00	00 65 02 f0 80 72 03 00	.1..... e....r..
0040	56 20 68 ad 71 74 34 cb	34 89 19 4d ae 03 0a d2	V h.qt4. 4..M....
0050	e6 f5 7c 5e c3 07 a9 89	a5 5d 31 b0 c2 23 42 80	.. ^....]1..#B.
0060	b8 fc 31 00 00 04 f2 00	00 00 0c 00 00 03 8f 34	14
0070	00 00 00 34 01 90 77 00	08 01 00 00 04 e8 89 69	Session ID
0080	00 12 00 00 00 00 89 6a	00 13 00 89 6b 00 04 00jk...
0090	00 00 03 00 00 00 72	03 00 00r

Type:Request SubType:SetVariable

Figure 5.4 S7CommPlus Function Request Packet

Figure 5.4 shows a S7CommPlus Connection packet. From the 5th to 37th bytes, is the encryption array. The 5th byte represented the Encryption length and the rest represented the Encryption Part which is calculated by TIA Portal. This Encryption Part will be talked in the next chapter.

6. Fun with the Encryption

In chapter 5, we found two encryptions in the S7CommPlus protocol packets, one in the second connection packet and the other in function packets. Using reverse debugging techniques, we found these encryption is calculated by TIA Portal through a file named OMSp_core_managed.dll. In this .dll file, TIA Portal generated the encryption parts using private algorithms.

6.1 Connection packet encryption

The Connection Encryption arrays in the Second connection packet send by TIA Portal are two 16 bytes' arrays. These two arrays are both calculated by OMSp_core_managed.dll.

In the first connection response packet, we have already known a random value array generated by the PLC with the length of 20. Using Windbgv6.1.12, we can find this value array is the input parameter for the first encryption of connection packet encryption. Figure 6.1 shows a first connection response packet send by the PLC. The Value Array is “0xc2, 0x11, 0x70, 0xdf, 0xd4, 0x03, 0x6c, 0xf1, 0x52, 0x9f, 0x47, 0x90, 0x1c, 0xd0, 0xca, 0xac, 0x63, 0x7f, 0xd5”. Figure6.2 shows a debugging procedure, we found that the eax+244 is “0x70, 0xdf, 0xd4, 0x03, 0x6c, 0xf1, 0x52, 0x9f, 0x47, 0x90, 0x1c, 0xd0, 0xca, 0xac, 0x63”. Compare to the first connection response packet, we found these arrays has the same value in the Value Array's 3rd to 17th bytes.

0030	20 00 df 31 00 00 03 00 00 c5 02 f0 80 72 01 00	..1.....r..
0040	b6 32 00 00 04 ca 00 00 00 01 36 00 02 87 53 87	.2..... ..6...S.
0050	4a a1 00 00 01 20 82 1f 00 00 a3 81 69 00 15 02	J.....i...
0060	30 31 a3 82 2b 00 04 82 80 80 00 a3 82 2d 00	01...+...-
0070	15 10 4f 4d 5:Value Array 15 4c 2e 37 30 37 30 2e	..OMSP.R EL.7070.
0080	31 34 a3 82 2f 10 02 14 c2 11 70 df d4 03 6c f1	14.../... .p...l.
0090	52 9f 99 47 90 1c d0 ca ac 63 7f d5 a3 82 32 00	R..G..... .c...2.
00a0	17 00 00 01 3a 82 3b 00 04 83 40 82 3c 00 04 83:.. ..@.<....
00b0	00 82 3d 00 04 84 80 c1 40 82 3e 00 04 84 80 c1	...=..... @.>.....
00c0	00 82 3f 00 15 1b 31 3b 36 45 53 37 20 32 31 34	..?...1; 6E57 214
00d0	2d 31 41 47 34 30 2d 30 58 42 30 20 3b 56 34 2e	-1AG40-0 XB0 ;V4.
00e0	31 82 40 00 15 05 32 3b 38 31 38 82 41 00 03 00	1.@...2; 818.A...
00f0	03 00 a2 00 00 00 72 01 00 00r ...

Figure 6.1 First S7CommPlus Connection Response Packet with Value Array

```

Disassembly
Offset: @Scopeip
0:024:x86> p
0030 75a2 jne    OMSp_core_managed+0x1dce6d (182cced9)
0034 0b5508 mov    edx,dword ptr [ebp+8]
0038 83c214 add    edx,14h
003c 52d0 push   edx
0040 245e0 lea    eax,[ebp-20h]
0044 50a0 push   eax
0048 84d08 mov    ecx,dword ptr [ebp+8]
004c 91c12c020000 add    ecx,22Ch
0050 51a0 push   ecx
0054 84dbffff call   OMSp_core_managed+0x1daa30 (182cced9)
0058 8340c add    edx,0Ch
005c 0b5508 mov    eax,dword ptr [ebp+8]
0060 81c22c020000 add    edx,22Ch
0064 52d0 push   edx
0068 84508 call   OMSp_core_managed+0x1dc810 (182cced9)
0072 82c404 add    eax,4
0076 84508 mov    eax,dword ptr [ebp+8]
007a 0b53844020000 movzx  eax,byte ptr [eax+44h] ds
0080 01ff000000 and    ecx,0FFh
0084 8b5508 mov    edx,dword ptr [ebp+8]
0088 0fb68245020000 movzx  eax,byte ptr [edx+245h]
0092 25ff000000 and    eax,0FFh
0096 c1e008 shr    eax,8
009a 0b58 mov    or    ecx,ecx
009e 8b5508 mov    edx,dword ptr [ebp+8]
00a2 0fb68246020000 movzx  eax,byte ptr [edx+246h]
00a6 25ff000000 and    eax,0FFh
00a8 c1e010 shr    eax,10h
00b2 0b58 mov    or    edx,edx
00b6 8b5508 mov    edx,dword ptr [ebp+8]
00b8 0fb68247020000 movzx  eax,byte ptr [edx+247h]
00bc 25ff000000 and    eax,0FFh
00c0 c1e018 shr    eax,18h

```

Figure 6.2 First encryption part in the second S7CommPlus Connection packet

With the value array as input, TIA Portal used a XOR (we call this Encryption1) to generated the first encryption part in the second S7CommPlus Connection packet:

Value Array + Encryption1 = Connection Encryption Part 1

Using the Connection Encryption Part 1 as input, TIA Portal continue its private algorithm which is more complex than a XOR (we call this Encryption2) to calculated the second encryption part in the second S7CommPlus Connection packet:

Connection Encryption Part 1 + Encryption2 = Connection Encryption Part 2

Figure 6.3 shows the result of Connection Encryption Part 1 and Connection Encryption Part 2 from the Windbg and the second S7CommPlus Connection packet.

Figure 6.3 Encryption part in the second S7CommPlus Connection packet

6.2 Function packet encryption

Each function packet send by the TIA Portal has a 32 bytes' array called Encryption Part. This array is calculated by OMSP_core_managed.dll.

Using Windbg, we found an array with Session ID in it, is the input parameter of Function packet encryption. Except the Session ID, the other value is constant, as Figure 6.4 shows.

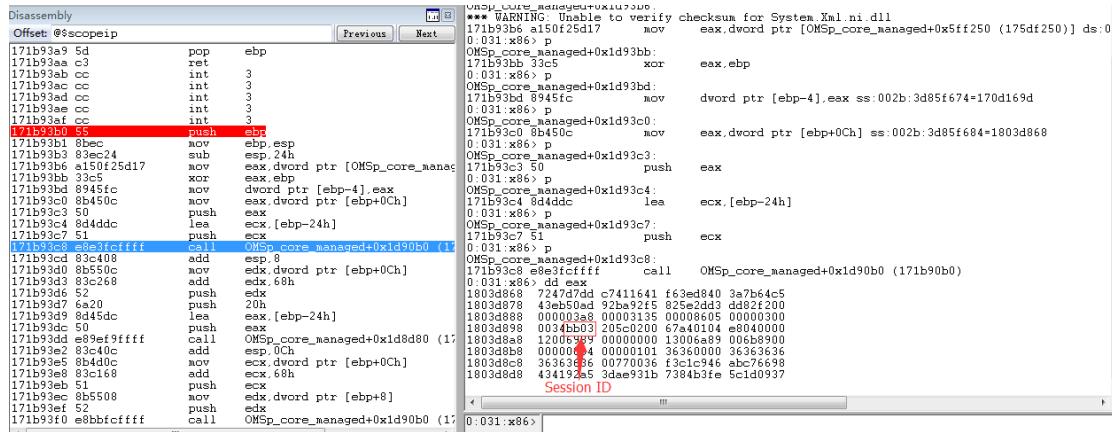


Figure 6.4 Input parameter for S7CommPlus Function packet encryption

TIA Portal used a complex algorithm (we call this Encryption3) to generate the Encryption Part of S7CommPlus Function packet:

Constant Array (with Session ID) + Encryption3 = Function Encryption Part

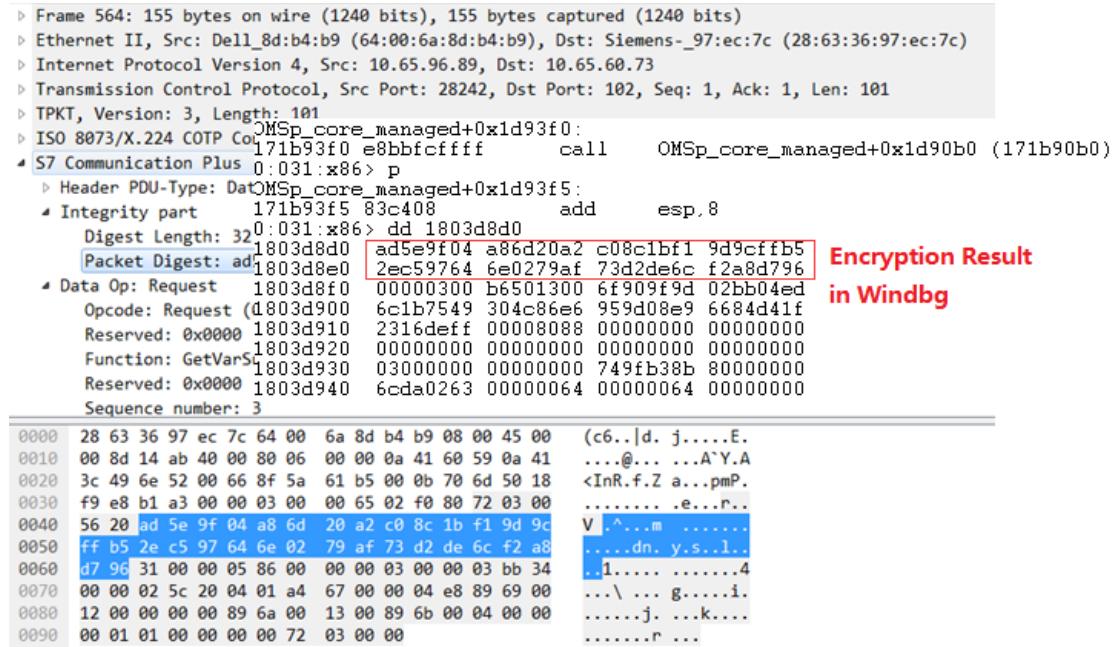


Figure 6.5 Function Encryption part in S7CommPlus Function packet

Figure 6.5 shows the result of Function Encryption Part from the Windbg and the S7CommPlus Function packet.

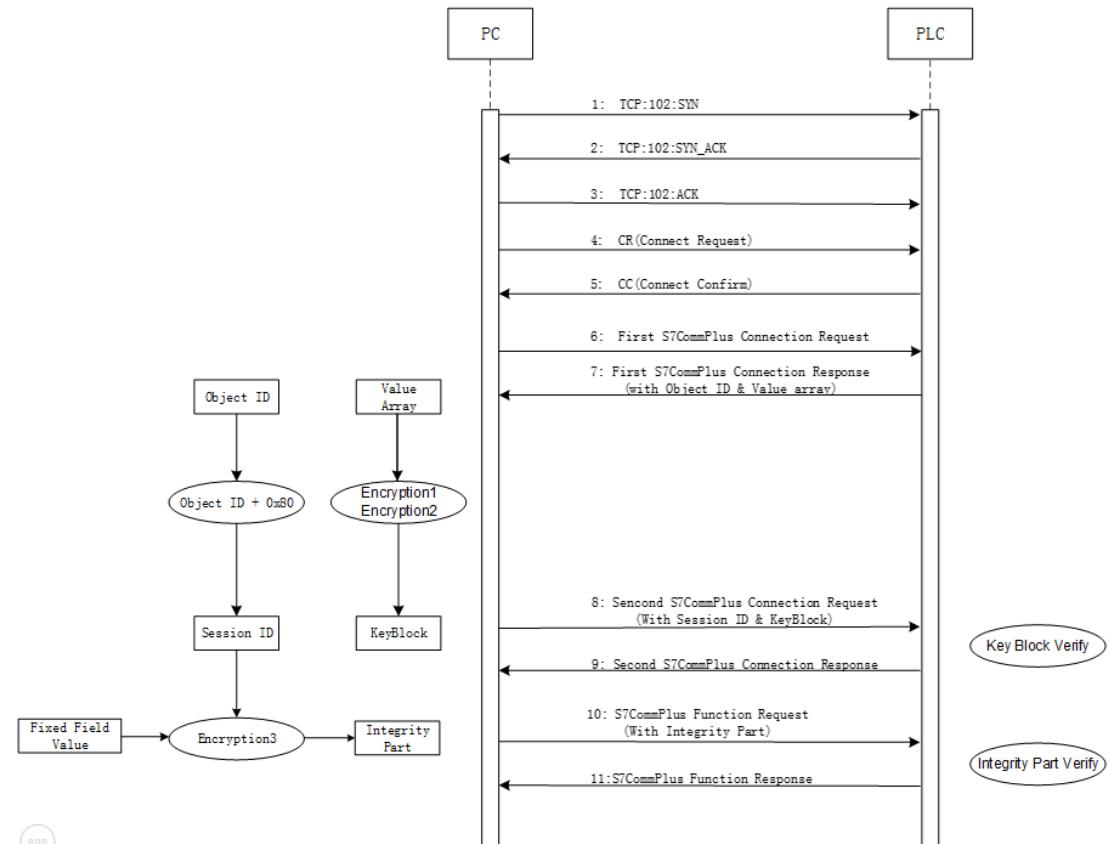
6.3 S7CommPlus Communication

Based on the research of S7CommPlus protocol encryptions above, we can get the S7CommPlus protocol communication sequence shown in figure 6.6. To establish a connection between the TIA Portal and PLC, the three-way handshake TCP connection has been used first. After the COTP connection

(CR & CC), TIA Portal will send an S7CommPlus Connection request. The first S7CommPlus Connection Response packet include an Object ID and a Value Array which is generated by the PLC. When receiving the Object ID and the Value Array, the Session ID and Key Block will be calculated by TIA Portal. Then, the second S7CommPlus Connection request packet including Session ID and Key Block will send to the PLC. If the Session ID and Key Block is correct, after the verify of PLC, a response packet will be send back to finish the S7CommPlus connection. Each S7CommPlus Function Request packet include an integrity part. The integrity part is calculated by TIA Portal using the Session ID and a fixed Field Value as its input parameter. When the PLC receives the S7CommPlus Function Request packet, the integrity part will be verified. The S7CommPlus Function Response packet could be send only when the verify was correct.

Figure 6.6 S7CommPlus protocol communication sequence with encryptions

7. Protections



7.1 Code level

Use code confusion techniques and anti-Debug techniques for the key DLL files like OMSp_core_managed.dll. Siemens didn't do any code protection to

the key DLL files. Therefore, it is very easy for attackers to debug and then find the encryption algorithm.

7.2 Design level

In the new S7CommPlus protocol, some complex encryption algorithm has taken by Siemens to against the replay attack. However, the input parameter and the encryption algorithm are not variable. We recommended to use a private key as an input parameter for encryption algorithm in the communication between Siemens software and PLCs.

7.3 Protocol level

Encrypt the whole packets instead of the key byte encryption.

8. Conclusion

In this paper, we found that the secure Siemens protocol still has the risk of being exploited. Using reverse debugging techniques, the encryption algorithm of TIA Portal for anti-replay attack can be break. Then, using replay attack, the PLC can be controlled. According to our research, some protections were proposed in code level, design level and protocol level.

REFERENCES

- [1] Ralf Spenneberg, Maik Brüggemann, Hendrik Schwartke
PLC-Blaster: A Worm Living Solely in the PLC. Black Hat 2016 USA
- [2] Dillon Beresford. Exploiting Siemens Simatic S7 PLCs. Black Hat 2011
USA
- [3] Thomas_v2. S7comm Wireshark dissector plugin.
<http://sourceforge.net/projects/s7commwireshark/files/>.