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.

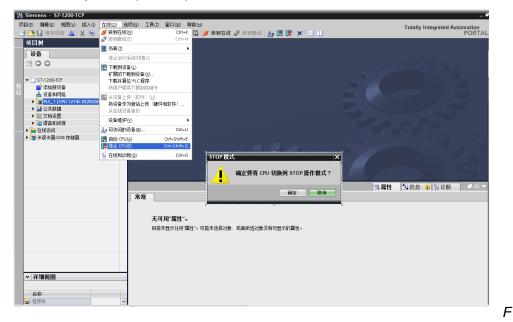
3.2 Siemens PLCs protocols

Siemens PLCs use a private protocol to communicate. It is a binary protocol utilizing both TPKT 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.



igure 3.1 TIA Portal CPU STOP

	2	99	inan i d								
				1 2 2	000 000						
S7-1200-TCP		1	名称	10 1	地址	显示格式	當視值	修改值	9	注释	
		4	TCON DI	RUCY	26231	布尔型	FALSE	IS \$2.12			
■ 添加新设备	' - '	+ 5	"TCON_DI			布尔型	TRUE	TRUE			^
		6	"TCON_DI			布尔型	FALSE	inoc.			
CC B 1001388 CC B 100138 CC B 1001388 CC B 1001388 CC B 100138 CC B 10013 CC B 1001 CC B 100 CC B 100		7	TCON DI			十六进制	16#0000				•
1 设备组态		8		DB*.BUSY		布尔型	FALSE				=
■ 成面相応 し 在线和诊断		9		DB".DONE		布尔型	FALSE				
 ▲ 程序块 		10		_DB".ERROR		布尔型	TRUE	TRUE			
 □ 4 中次 ■ 工艺対象 		11	"TCON_DI			布尔型	FALSE				
 ▶ → 小部原文件 		12		DB".REQ		布尔型	FALSE				
▶ □ PLC 变量		13	"DI00"		%10.0	布尔型	FALSE	TRUE			
▶ Co PLC 数据类型		14	"DI01"		\$610.1	布尔型	FALSE	TRUE			
▼ □□ 监控与强制表		15		onnection_D		无符号十进制					
■ 添加新监控表		16	"A001"	_	%QW12			0			
		17 🗙	*DB3*.AM	01	D85.777		3	10#40	A -		
		18	*DO06*		%Q0.6	布尔型	FALSE		Ä		
記録組ま		19	"AI04"		%IW116			67	🗹 🔺		
Traces		20			<添加>						
24 程序信息											
▶ 3 设备代理数据		<									~
■ 文本列表	1 H								-	1 m	
▶ □ 本地視块									🧟 属性	🔄 信息 🔒 🗓 诊断	
▶ 🛅 分布式 I/O		设备	信息 首	接信息	报警显	示					
▶ → 公共数据	11		备出现问题								
 			曲山沈門ᡂ			报警	详细信息		凝結		
▼ 🐻 语言和资源			1981 F	校 面/1長校 PLC_1	c .	(現書) 错误		1. 请参见设备诊断。	7		
▶ 项目语言	1	捕扶	NON	rec_r		相庆	史沙伯大中和国地	1. 開始外的改革後期。			
■ 项目文本	\sim										
114 Amilia (a)	100										
计组代图	-										
 □ 坝目火本 > 详细视图 名称 	~										

igure 3.2 TIA Portal value monitor and modify

F

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.	Tine	Source	Destination	Protocol	Length Info
-	1019 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 : 37:26.266384	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 13:37:26.266509	10.65.96.89	10.65.60.73	TCP	54 5208→102 [ACK] Seg=1 Ack=1 Win=64240 Len=0
	1023 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
	1007P Connection .: 37:26.269514	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 ←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 →5208 PDU-Type: [Connect] Op: [Response] Function: [CreateObject] S
	1(S7CommPlus 13:37:26.287630	10.65.96.89	10.65.60.73	COTP	61 DT TPDU (0) [COTP fragment, 0 bytes]
	16Connection 13:37:26.331976	10.65.96.89	10.65.60.73	S7COMM-PLUS	472 +5208 PDU-Type: [Data] Op: [Request] Function: [SetMultiVariables]
	1039 2017-02-24 13:37:26.360397	10.65.60.73	10.65.96.89	TCP	60 102→5208 [ACK] Seq=233 Ack=696 Win=8192 Len=0
	1054 2017-02-24 13:37:26.459946	10.65.60.73	10.65.96.89	S7COMM-PLUS	86 →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 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=703 Win=8192 Len=0
	1092 2017-02-24 13:37:26.693001	10.65.96.89	10.65.60.73	S7COMM-PLUS	155 +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 →5208 PDU-Type: [DataFW1_5] Op: [Response] Function: [GetVarSubStre
	1094 2017-02-24 13:37:26.697987	10.65.96.89	10.65.60.73	COTP	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 +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 →5208 PDU-Type: [DataFW1_5] Op: [Response] Function: [SetVariable]
	S7CommPlus Function 27.087691	10.65.96.89	10.65.60.73	COTP	61 DT TPDU (0) [COTP fragment, 0 bytes]
	:Stop PLC 27.157371	10.65.60.73	10.65.96.89	TCP	60 102+5208 [ACK] Seg=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 +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 →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.

								Se	que	nce	2			PI	DU	Тур	e
		Тур	e	S	ub-	Ty	pe	N	umb	er			rote	-	-	in the second	Data Length
0030	fa	cd	b2	29	00	00	03	00	00	eb	02	fØ	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	iServerS
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	151::: 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	6e	65	63	74	69	6f	6e	20	49	32	31	37	2d	4c	onnectio n I217-L
0060	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	а3	82	LEI-PC_1 859921
00e0	26	00	04	01	a3	82	2c	00	12	01	c9	c3	80	а3	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	ntainerr
0120	00	F	ran	1e	Bou	inc	lan	V									

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.

	Туре	:Re	spo	nse								c	bje	ect	ID	
0030	20 0	07	05	c 00								80				.p\
0040	b6 3	20	00	0 04	са	00	00	00	01	36	00	02	87	0f	87	.26
0050	1a a:	10	00	0 01	20	82	1f	00	00	a3	81	69	00	15	02	i
0060	30 33	1 a	38	2 2b	00	04	82	80	80	80	00	a3	82	2d	00	01+
0070	15 10	04	f 4	d 53	50	2e	52	45	4c	2e	37	30	37	30	2e	OMSP.R EL.7070.
0800	31 34															14/0.
0090	2d d															<4 ;aV2.
00a0	17 0 00 8	0 0	00	1 3a	82	71-	00		07	10	01	3c	00	04	83	;@.<
00c0	00 8	23	f 0	0 15	1Ł\	/alı	ie A	rray				20	32	31	34	?1; 6ES7 214
00d0	2d 3:															-1AG40-0 XB0 ;V4.
00e0	31 8	24	0 0	0 15	05	32	3b	38	31	38	82					1.@2; 818.A
00f0	03 0	0 a	20	0 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.

_			
	000	IOF	N ID
	e 53	101	

0030	fa	08	b2	e0	00	00	03	00	01	a2	02	f0	80	72	02	01	r
0040	93	31	00	00	05	42	00	00	00	02	00	00	03	8f	34	00	.1B4.
0050	00	03	8f	02	02	8e	26	82					00				&. 2
0060	8e	09	00	04	00	8e	0a	00	02	00	8e	0b	00	17	00	00	
0070	07	21	8e	22	00	05	de	d0	cd	b0	с8	fc	90	f3	1a	8e	.1."
0080	23	00	04	84	82	10	8e	24	00	04	00	00	8e	0c	00	17	#\$
0090	00	00	07	21	8e	22	00	05	c 1	e5	ba	f1	82	a4	a1	dc	
00a0	ec	8e	23	00	04	84	82	01	8e	24	00	04	00	00	8e	0d	#\$
00b0	00	14	00	81	34	ad	de	e1	fe	b4	00	00	00	01	00	00	4
00c0	00	01	00	00	00	ec	dc	10	49	10	d7	95	83	01	01	00	I
00d0	00	00	00	00	00	1a	73	0 8	1f	0 9	6b	42	bd	10	01	00	skB
00e0	00	00	00	00	00	01	99	ec	e4	62	a6	13	5c	ас	6f	d5	b\.o.
00f0	bf	fa	d9	85	44	bd	b0	11	80	6c	95	91	9b	e9	f8	ed	D1
0100	60	55	35	97	3e	5a	f6	0c	fb	85	57	8b	42	47	f2	7f	`U5.>ZW.BG
0110	d6	8b	1b	e!	inet	Col		etio	n E	nen	ti	ion	3e	67	2f	45	n.Ha.>g/E
0120	f9	53	59	75	e/	au	21	ctio	20	40		41	08	3b	bb	22	.SYu?{ &F.0.."
0130								f7					42				BHUB\
0140	f7	ff	66	bf	3f	1d	4b	2d	52	b2	1a	87	4b	6e	2c	13	f.?.K- RKn,.
0150	4c	85	20	bf	55	9c	2d	7e	с8	01	ce	62	94	44	bd	8a	LU~b.D
0160								66	82	00	02	00	17	00	00	01	zotf
0170	3a	82	3b	00	04	83	22	<u> </u>	2		23	27	- 22	~~	21	<u> 10</u>	:.;=.
0180	04	84	80	c1	00	82	ec.	ond	<u> </u>	une Une	cui	201	:nc	гур		J 0	
0190	15	00	82	40	00	15	1a	31	3b	36	45	53	37	20	32	31	6ES7 21;6ES7 21
01a0	34	2d	31	41	47	34	30	2d	30	58	42	30	3b	56	34	2e	4-1AG40- 0XB0;V4.
01b0	30	82	41	00	03	00	03	00	00	00	00	04	e8	89	69	00	0.Ai.
01c0	12	00	00	00	00	89	6a	00	13	00	89	6b	00	04	00	00	jk
01d0	00	00	00	00	72	02	00	00									r

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:

Session ID = 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.

E	ncry	/pti	on	len	gth	1		Enc	rypt	tior	n Pa	nt					
0030	f6	6c	b1	a3	00	00	03	00	00	65	02	f0	80	72	03	00	.ler
0040	56	20	68	ad	71	74	34	cb	34	89	19	4d	ae	03	0a	d2	V h.qt4. 4M
0050																	
0060	b8	fc	31	00	00	04	f2	00	00	00	0c	00	00	03	8f	34	1 4
0070	00	00	00	34	01	90	77	00	08	01	00	00	04	e8	89	69	Session ID i
0800	00	12	00	00	00	00	89	6a	00	13	00	89	6b	00	04	00	jk
0090	00	00	03	00	00	00	00	72	03	00	00						r
	_		_														

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, 0x9f, 0x47, 0x90, 0x1c, 0xd0, 0xca, 0xac, 0x63, 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	1r
0040	b6	32	00	00	04	са	00	00	00	01	36	00	02	87	53	87	.26S.
0050	4a	a1	00	00	01	20	82	1f	00	00	a3	81	69	00	15	02	Ji
0060	30	31	a3	82	2b	ØЙ	Ø 4	82	80	80	80	00	a 3	82	2d	00	01+
0070	15	10	4f	4d	5:V	/alu	e A	rray									OMSP.R EL. 7070.
0080	31	34	a3	82	2f	10	02	14	с2	11	70	df	d4	03	6c	f1	14/ <mark>p1.</mark>
0090	52	9f	99	47	90	1c	d0	са	ас	63	7f	d5	a3	82	32	00	RGc <mark>2.</mark>
00a0	17	00	00	01	Зa	82	Зb	00	04	83	40	82	3с	00	04	83	:.;@.<
00b0	00	82	3d	00	04	84	80	c1	40	82	3e	00	04	84	80	c 1	= @.>
00c0	00	82	3f	00	15	1b	31	3b	36	45	53	37	20	32	31	34	?1; 6ES7 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	00	72	01	00	00						r

Figure 6.1 First S7CommPlus Connection Response Packet with Value Array

			L3-002J	55-002D C	15-002D C	55-00ZD 1	.s-0000 ys-
Disassembly		I.I. 🖾		_managed+0	x1dcefa:		
Offset: @\$scopeip		Previous Next	182ccefa		mov	eax.	dword ptr [
182ccec9 75a2		OVC=	0:024:x86	>p			
182ccecb 8b5508	jne	OMSp_core_managed+0x1dce6d (18					
182ccece 83c214	mov add	edx,dword ptr [ebp+8] edx.14h		Ofb6884402	0000 mov	zx ecx.	byte ptr [e
			0:024:x86				o = oo
182cced1 52 182cced2 8d45e0	push	edx .	3e14e5b0	00000001			
	lea	eax,[ebp-20h]	3e14e5c0	17df3b51			
182cced5 50	push	eax	3e14e5d0	fd7c9c68			
182cced6 8b4d08	mov	ecx,dword ptr [ebp+8]	3e14e5e0	febaa0dd			
182cced9 81c12c020000	add	ecx, 22Ch	3e14e5f0	2bc73f83			
182ccedf 51	push	ecx	3e14e600	9058098Ъ			
182ccee0 e84bdbffff	call	OMSp_core_managed+0x1daa30 (18		с7Ь29968			
182ccee5 83c40c	add	esp,OCh	3e14e620	ca0194c3	8848dfba	ae9dcff6	a32ec25d
182ccee8 8b5508	MOV	edx,dword ptr [ebp+8]	0:024:x86				
182cceeb 81c22c020000	add	edx,22Ch	3e14e5b0	00000001			
182ccef1 52	push	edx	3e14e5c0	17df3b51			
182ccef2 e819f9ffff	call	OMSp_core_managed+0x1dc810 (18		fd7c9c68			
182ccef7 83c404	add	esp,4	3e14e5e0	febaa0dd			
182ccefa 8b4508	MOV	eax,dword ptr [ebp+8]	3e14e5f0	2bc73f83			
182ccefd 0fb68844020000	MOVZX	ecx, byte ptr [eax+244h] ds:	3e14e600	9058098Ъ	85fbd211	ec75a6eb	57ea90e3
182ccf04 81e1ff000000	and	ecx,0FFh	3e14e610	c7b29968	42494Ъ79	5a599d48	0db30dab
182ccf0a 8b5508	MOV	edx,dword ptr [ebp+8]	3e14e620	ca0194c3	8848dfba	ae9dcff6	a32ec25d
182ccf0d 0fb68245020000	MOVZX	eax, byte ptr [edx+245h]	0:024:x86	> dd $eax+2$	44		
182ccf14 25ff000000	and	eax, OFFh	3e14e7f4	03d4df70	9f52f16c	1c904799	63accad0
182ccf19 c1e008	shl	eax,8	3e14e804	00000000	00000000	00000000	00000000
182ccf1c Obc8	or	ecx,eax	3e14e814	9d9a5ef8	f3e19f57	3ca5c89e	17df3b51
182ccf1e 8b5508	MOV	edx,dword ptr [ebp+8]	3e14e824	3b34bdf0	e7c33eaf	794b913d	2fbe76a2
182ccf21 0fb68246020000	MOVZX	eax, byte ptr [edx+246h]	3e14e834	a6aee308	1422a1f8	45ee59a3	38614df3
182ccf28 25ff000000	and	eax,0FFh	3e14e844	76697be0	cf867d5e	f297227b	5f7ced44
182ccf2d c1e010	shl	eax,10h	3e14e854	ebf32518	3c67e209	ce32eae5	48a3d615
182ccf30 0bc8	or	ecx,eax	3e14e864	4d5dc610	284543f1	8bdcb346	70c29be6
182ccf32 8b5508	MOV	edx.dword ptr [ebp+8]					
182ccf35 0fb68247020000	MOVZX	eax, byte ptr [edx+247h]	1				
182ccf3c 25ff000000	and	eax, OFFh				"	
182ccf41 c1e018	shl	eax,18h	0:024:x86	>			
<							
· L		r		1			

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.

0030	fa 08 b2 e0 00 00 03 00 01 a2 02 f0 80 72 02 01r.
0040	93 31 00 00 05 42 00 00 00 02 00 00 03 d3 34 00 .1B4.
0050	00 03 d3 02 02 8e 26 82 32 01 00 17 00 00 07 08&. 2
0060	8e 09 00 04 04 00 8e 00 00 00 00 00 00 00 00 00 00 00 00 00
0070	07 21 8e OMSp_core_managed+0x1dd056: 182cd056 83c40c add esp.0Ch
0080	23 00 04 0:024 x86> dd 1913703c
0090	
00a0	ec 8e 23 1913705c 00000000 00000000 00000000 00000000 0000
0060	00 14 00 1913707c 8000000 006f 00f 0075006d 0069006e
00c0	00 01 00 1913708c 00610063 006907a 006e006f 003a0065 00 01 00 1913709c 00540020 00610072 0073006e 007a0061
00d0	00 00 00 ^{191370ac} 006f0069 0065006e 00440020 00540050
00e0	00 00 00
00f0	bf fa d90:024:x86>
0100	60 55 35
0110	d6 8b 1b e ⁵ First Encryption Part 61 9b 3e 67 2f 45n.Ha.>g/E
0120	f9 53 59 75 e/ ad 3f /b 26 46 8f 4f 08 3b bb 22 .SYu?{ &F.O.."
0130	cb e4 f6 25 ff 6b 16 88 fe 70 d4 11 ff 59 c0 cb%.kpY
0140	f5 ff 66 bf 3f 1d 4b 2d 52 b2 1a 87 4b 6e 2c 13f.?.K- RKn,.
0150	4c 85 20 bf 55 9c 2d 7e c8 bd 85 36 f3 f5 a9 bc LU~6
0160	78 8d 94 24 c7 d2 c3 8b 1d 00 02 00 17 00 00 01 x\$
0170	3a 82 3b 00 C1 02 00 02 2,00 C1 03 00 82/Bd 00 :.;
0180	04 84 80 c1 Second Encryption Part 1 00 87 1f 00
0190	1[0:024:x86> p 45 4 24 21 26 45 52 27 20 12 24 0 4 6557 24
01a0	340MSp core managed+0x1dd615
01b0	3(182cd615 83c40c add esp.0Ch
01c0	1 3e14e550 f33685bd 78bca9f5 c724948d 1d8bc3d2 Second Encryption
01d0	0(3e14e560 00000000 55572085 c87e2a9c 00000000 Calculated using Windbg
	3e14e580 3e14e5b0 19137064 3e14f814 00000000
	3e14e590 3e14f818 182c73c0 3e14e5b0 1913705c 3e14e5a0 3e14f814 00000000 00000000 00000000
	3e14e5b0 00000001 9d9a5ef8 f3e19f57 3ca5c89e
	3e14e5c0 17df3b51 00000004 1eb3fd9a 01cfdc35 0.024.9265 b1

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.

Disassembly		11 B	www.WARNING: Unable to verify checksum for System.Xml.ni.dll
Offset: @\$scopeip		Previous Next	171b93b6 a150f25d17 nov eax,dword ptr [OMSp_core_managed+0x5ff250 (175df250)] ds:0
			0:031:x86> p
171b93a9 5d	pop	ebp	OMSp_core_nanaged+0x1d93bb:
171b93aa c3	ret	2	171b93bb 33c5 xor eax.ebp
171b93ab cc 171b93ac cc	int	3 3 3	0:031:x86> p
171b93ac cc	int int	3	OHSp_core_managed+0x1d93bd: 171b93bd 8945fc mov dword ptr [ebp-4] eax ss:002b:3d85f674=170d169d
171b93ae cc	int	3	171b93bd 8945fc mov dword ptr [ebp-4],eax ss:002b:3d85f674=170d169d 0:031:x86> p
171b93af cc	int	3	0/031/x86> p ONSp_core_nanaged+0x1d93c0:
17159350 55	push	ebp	0755_077managed+08107300; 17159300 80450c nov eax.dword ptr [ebp+0Ch] ss:002b:3d85f684=1803d868
171b93b1 8bec	BOV	ebp,esp	0:031:x86> p
171b93b3 83ec24	sub	esp,24h	OMSp core managed+0x1d93c3:
171b93b6 a150f25d17	BOV	eax, dword ptr [OMSp_core_manag	171b93c3 50 push eax
171b93bb 33c5	xor	eax, ebp	0:031:x86> p
171b93bd 8945fc	ROV	dword ptr [ebp-4],eax	OMSp_core_managed+0x1d93c4:
171b93c0 8b450c	ROV	eax,dword ptr [ebp+0Ch]	171b93c4 8d4ddc lea ecx, [ebp-24h]
171b93c3 50	push	eax	0:031:x86> p
171b93c4 8d4ddc	lea	ecx,[ebp-24h]	OMSp_core_managed+0x1d93c7:
171b93c7 51	push	ecx	171b93c7 51 push ecx
171b93c8 e8e3fcffff	call	OMSp_core_managed+0x1d90b0 (13	0:031:x86> p
171b93cd 83c408 171b93d0 8b550c	add	esp,8	ONSp_core_managed+0x1d93c8:
171b93d0 8b550c	nov add	edx,dword ptr [ebp+0Ch] edx.68h	171b93c8 e8e3fcffff call OMSp_core_managed+0x1d90b0 (171b90b0) 0:031:x86> dd eax
171b93d6 52	push	edx	1803d88 7247d7dd c7411641 f63ed840 3a7b64c5
171b93d7 6a20	push	20h	1003d000 /24/0/dd C/41041 103E040 32/04C5
171b93d9 8d45dc	lea	eax,[ebp-24h]	18034888 000003a8 00003135 0000605 00000300
171b93dc 50	push	eax	1803d898 0034bb03 205c0200 67a40104 e8040000
171b93dd e89ef9ffff	call	OMSp core managed+0x1d8d80 (1)	1803d8a8 12005989 0000000 13006a89 006b8900
171b93e2 83c40c	add	esp, OCh	18034858 00000 4 00000101 36360000 36363636
171b93e5 8b4d0c	ROV	ecx, dword ptr [ebp+0Ch]	1803d8c8 36363636 00770036 f3c1c946 abc76698
171b93e8 83c168	add	ecx,68h	1803d8d8 434192a5 3dae931b 7384b3fe 5c1d0937
171b93eb 51	push	ecx	Session ID
171b93ec 8b5508	nov	edx.dword ptr [ebp+8]	
171b93ef 52	push	edx	
171b93f0 e8bbfcffff	call	OMSp_core_managed+0x1d90b0 (1)	0:031:x86>

Figure 6.4 Input parameter for S7CommPlus Function packet encryption

TIA Portal used a complex algorithm (we call this Encryption3) to generated the Encryption Part of S7CommPlus Function packet: Constant Array (with Session ID) + Encryption3 = Function Encryption Part

Frame 564: 155 bytes on wire (1240 bits), 155 bytes captured (1240 bits)

Ethernet II, Src: Dell_8d:b4:b9 (64:00:6a:8d:b4:b9), Dst: Siemens-_97:ec:7c (28:63:36:97:ec:7c) > Internet Protocol Version 4, Src: 10.65.96.89, Dst: 10.65.60.73 > Transmission Control Protocol, Src Port: 28242, Dst Port: 102, Seq: 1, Ack: 1, Len: 101 > TPKT, Version: 3, Length: 101 > ISO 8073/X.224 COTP ColDMSp_core_managed+0x1d93f0: > S7 Communication Plus 0:031:x86> p > Norder 200 Trace Device Plus 0:031:x86> p call OMSp_core_managed+0x1d90b0 (171b90b0) S/ Communication Pius 0:031:x86> p
> Header PDU-Type: DatDMSp_core_managed+0x1d93f5:
Integrity part 171b93f5 83c408 add esp.8
Digest Length: 32
0:031:x86> dd 1803d8d0
Packet Digest: ad
1803d8d0 ad5e9f04 a86d20a2 c08c1bf1 9d9cffb5
2ec59764 6e0279af 73d2de6c f2a8d796
Data 0p: Request 1803d8f0 0000030 b6501300 6f909f9d 02bb04ed
Opcode: Request (1803d900 6c1b7549 304c86e 959d08e9 6684d41f
Reserved: 0x0000 1803d910 2316deff 00008088 00000000 00000000 **Encryption Result** in Windbg
 Opcode:
 Redeved
 (1803d910
 2316deff
 00000000
 00000000
 00000000

 Function:
 GetVarS1803d920
 00000000
 00000000
 00000000
 00000000
 00000000

 Reserved:
 0x0000
 1803d940
 6cda0263
 0000000
 749fb38b
 80000000
 Sequence number: 3 0000 28 63 36 97 ec 7c 64 00 6a 8d b4 b9 08 00 45 00 (c6..|d. j....E@... ...A`Y.A <InR.f.Z a...pmP. 0010 00 8d 14 ab 40 00 80 06 00 00 0a 41 60 59 0a 41 0020 3c 49 6e 52 00 66 8f 5a 61 b5 00 0b 70 6d 50 18 0030 f9 e8 b1 a3 00 00 03 00 00 65 02 f0 80 72 03 00r...e....r.. V 0040 56 20 ad 20 a2 0050 ff b5 2e c5 97 64 6e 02 79 ar 75 02 0e ce re d d7 96 31 00 00 05 86 00 00 00 03 00 00 03 bb 34 0060 <mark>...</mark>1......4 0070 00 00 02 5c 20 04 01 a4 67 00 00 04 e8 89 69 00 ...\ ... g....i. 0080 12 00 00 00 00 89 6a 00 13 00 89 6b 00 04 00 00j.k.... 0090 00 01 01 00 00 00 00 72 03 00 00r ...

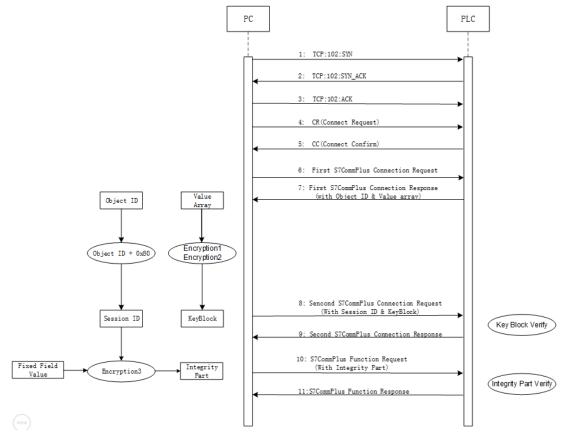
Figure 6.5 Function Encryption part in S7CommPlus Function packet Figure 6.5 shows the result of Function Encryption Part from the Windbg and the S7CommPlus Function packet.

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/.