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

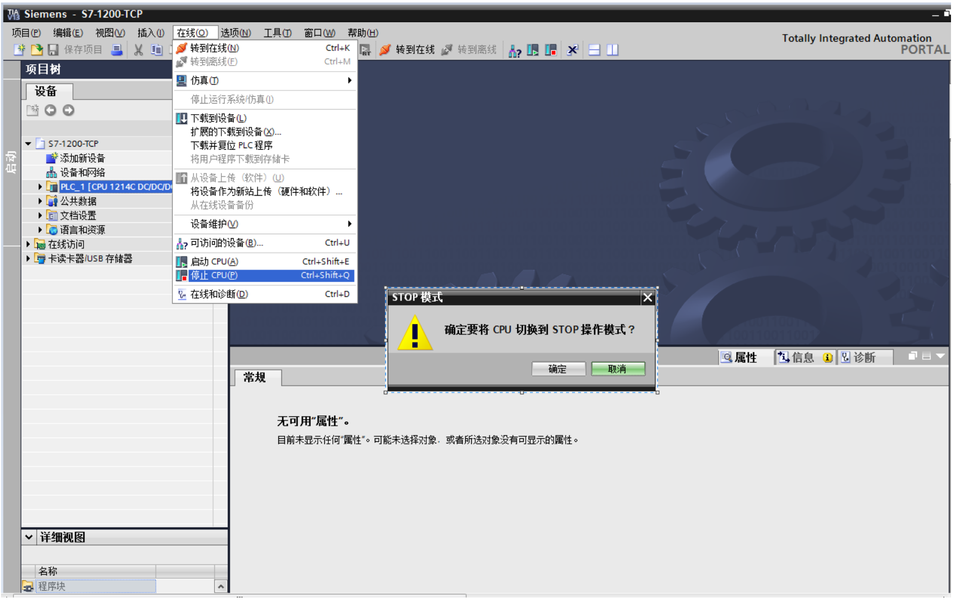
Programmable Logic Controllers (PLC) is responsible for process control in industrial control system. A PLC contains a Central Processing Unit (CPU), some digital/analog inputs and outputs modules, communication module and some process modules like PID. Engineers programed user programs for automated 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.

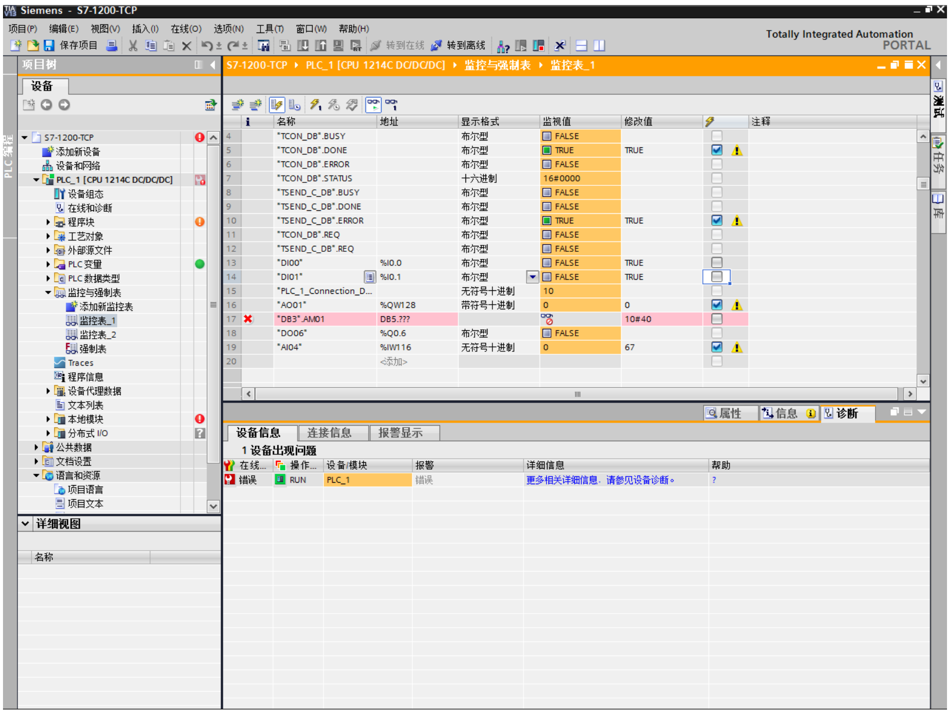
3.3 TIA Portal

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.



F

igure 3.1 TIA Portal CPU STOP



F

igure 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.

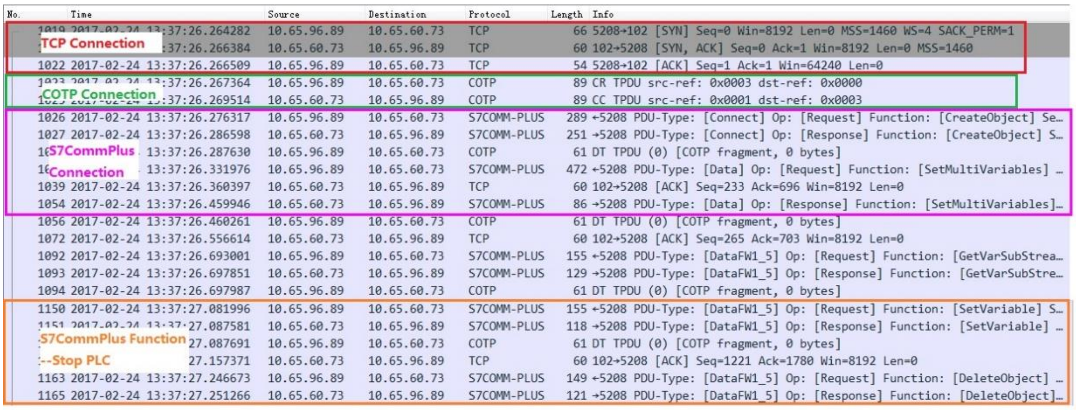


Figure 4.2 Stop PLC communication sequence

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.

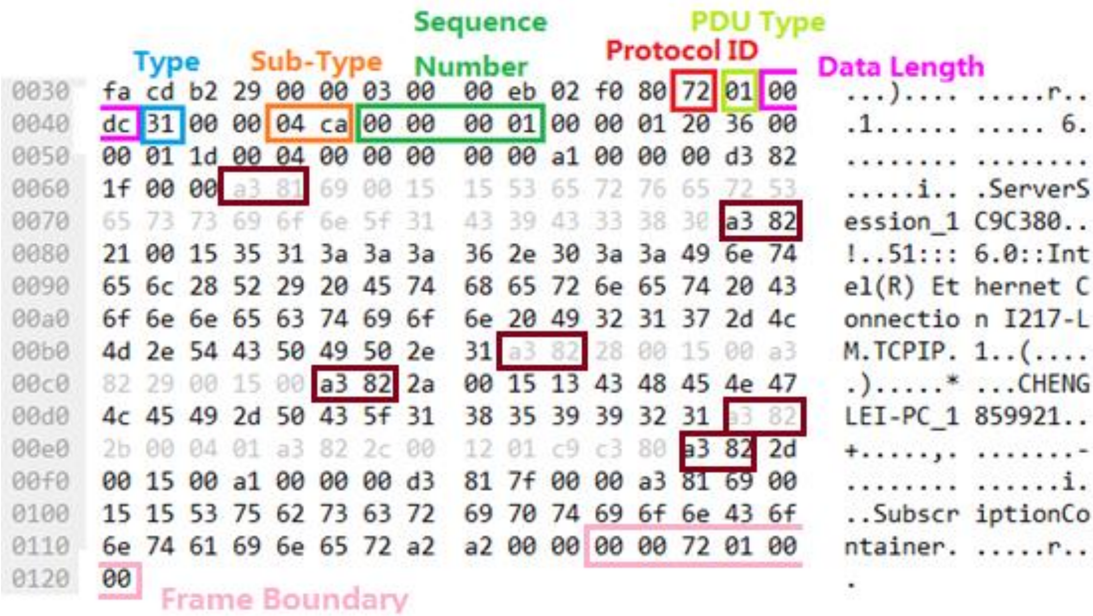


Figure 5.1 First S7CommPlus Connection Request Packet

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.

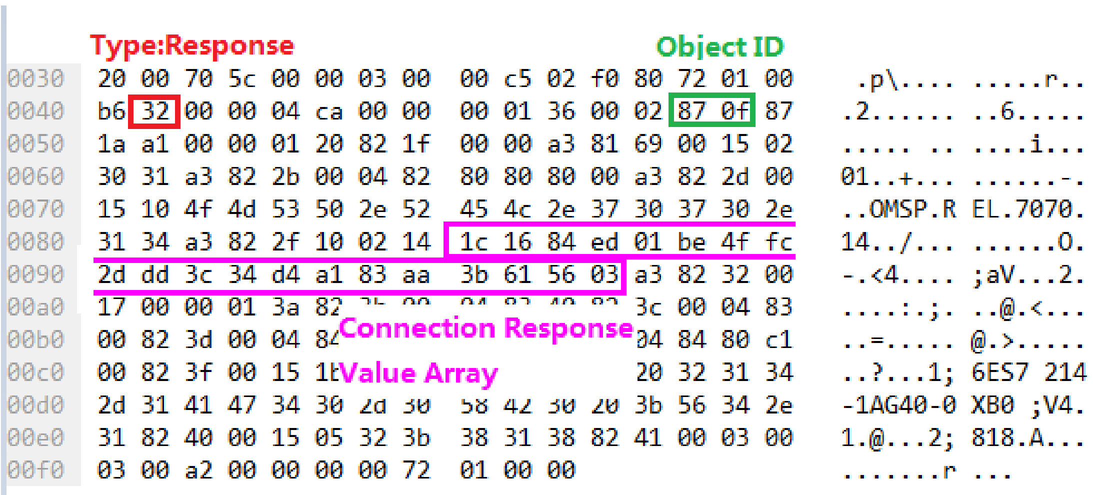


Figure 5.2 First S7CommPlus Connection Response Packet

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.

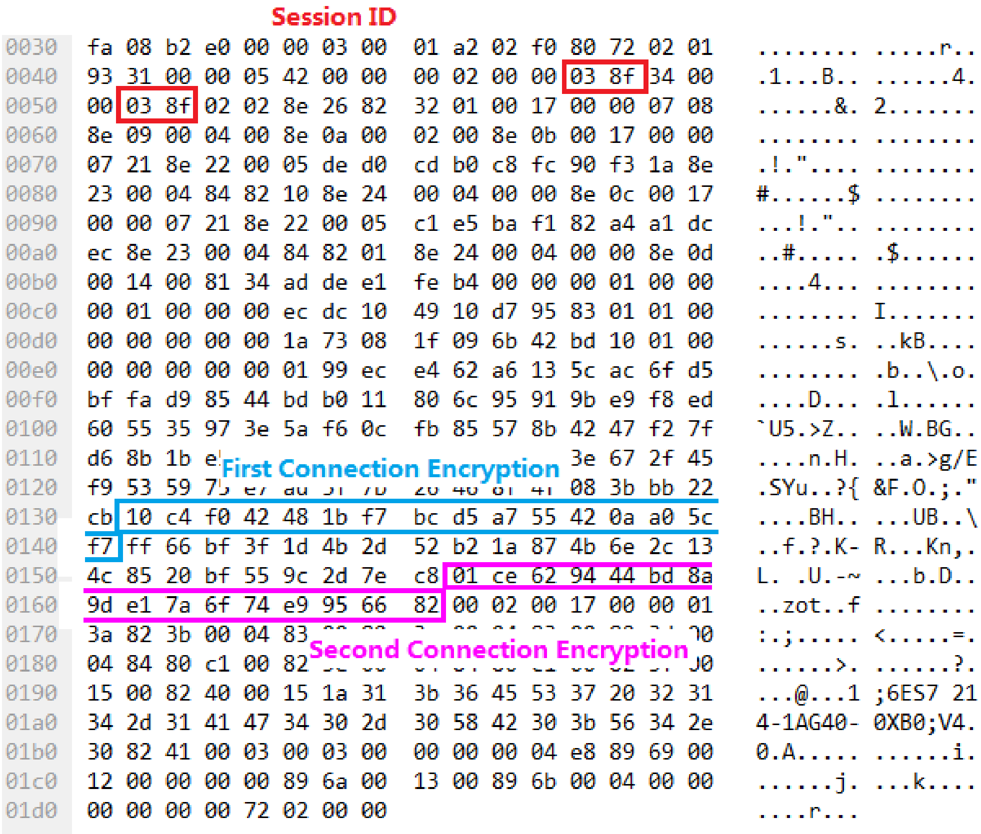


Figure 5.3 Second S7CommPlus Connection Request Packet

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.

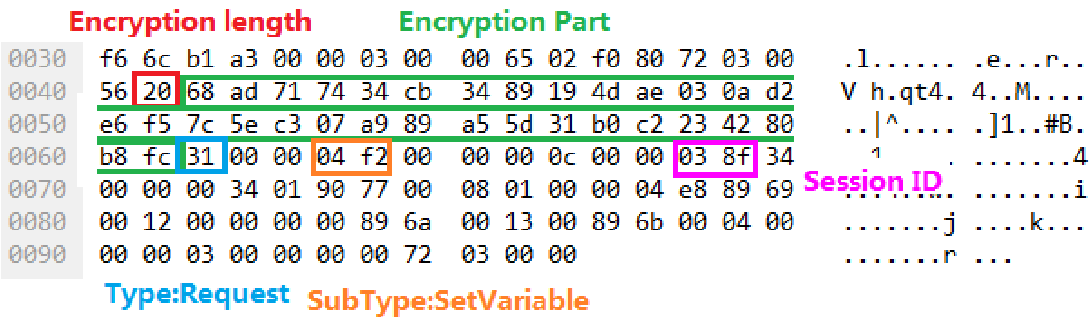


Figure 5.4 S7CommPlus Function Request Packet

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.

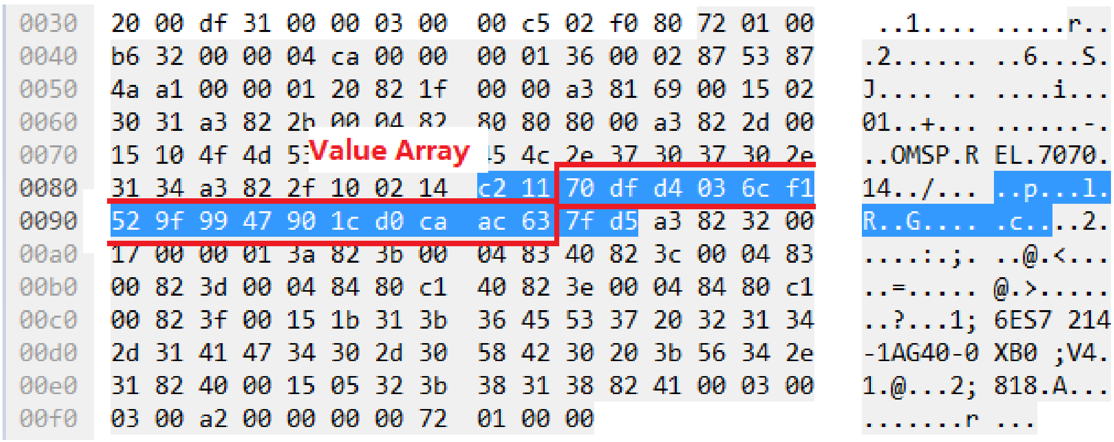


Figure 6.1 First S7CommPlus Connection Response Packet with Value Array

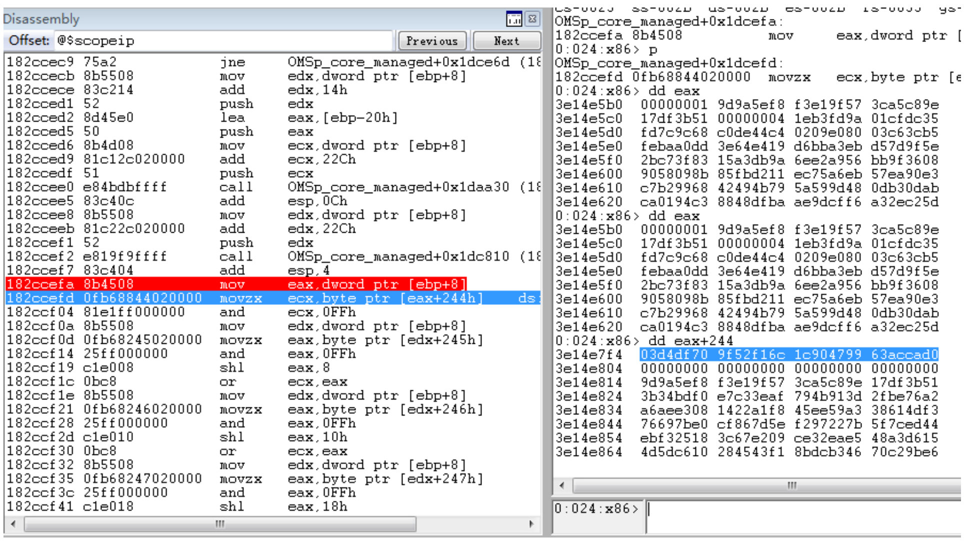


Figure 6.2 First encryption part in the second S7CommPlus Connection packet

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 2 + Encryption2 = Connection Encryption Part 2

Figure6.3 shows the result of Connection Encryption Part 1 and Connection Encryption Part 2 from the Windbg and the second S7CommPlus Connection packet.

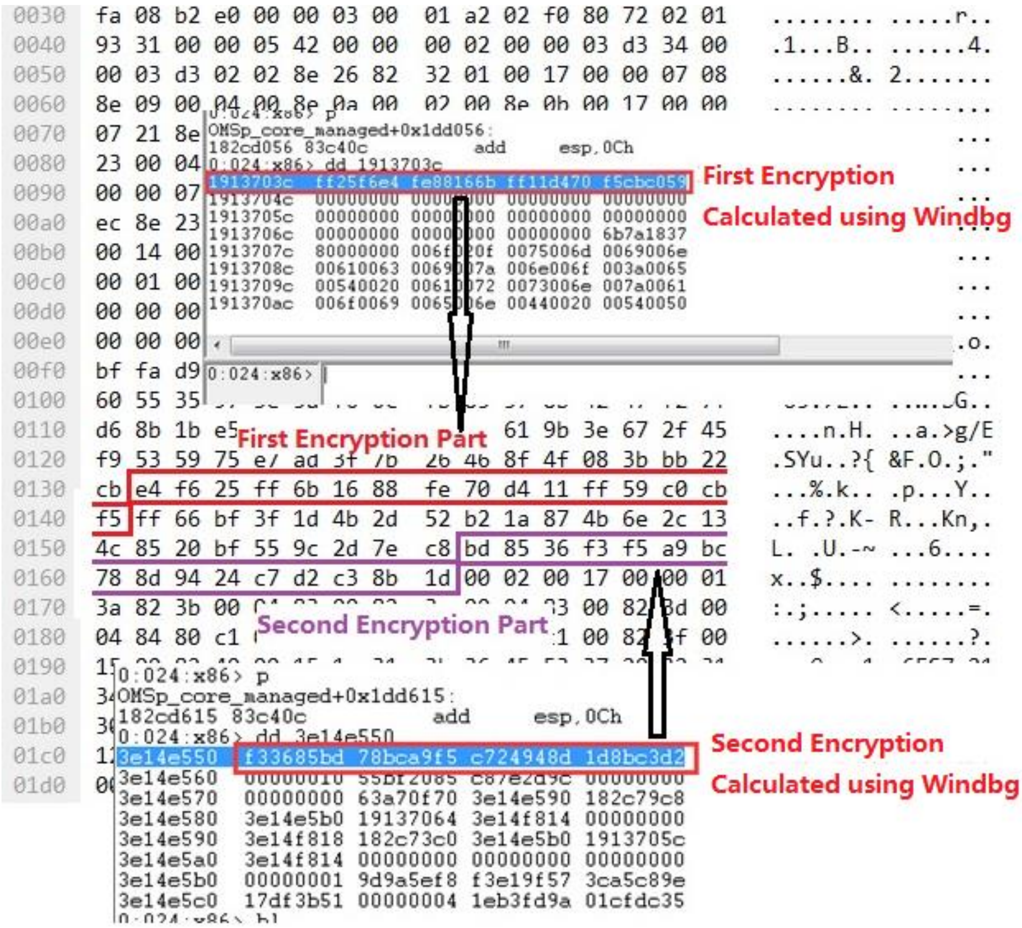


Figure 6.3 Encryption part in the second S7CommPlus Connection packet

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.

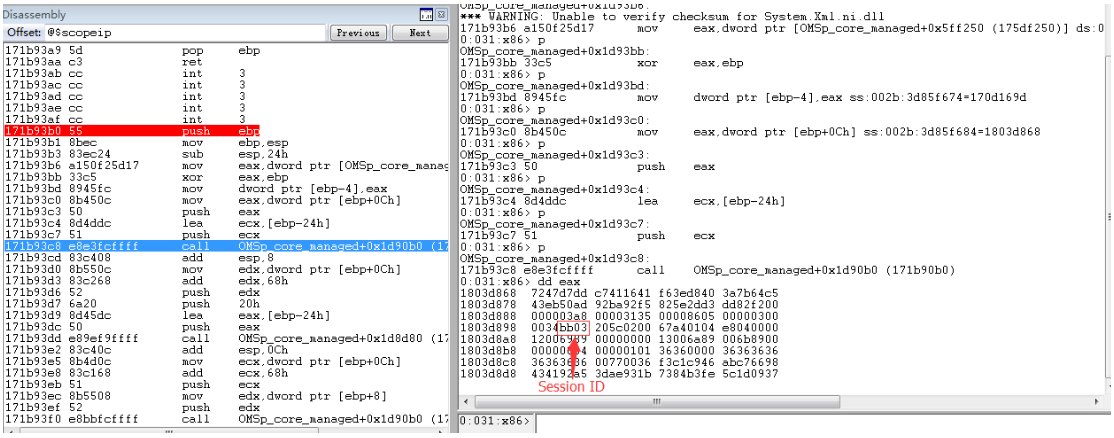


Figure 6.4 Input parameter for S7CommPlus Function packet encryption

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

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

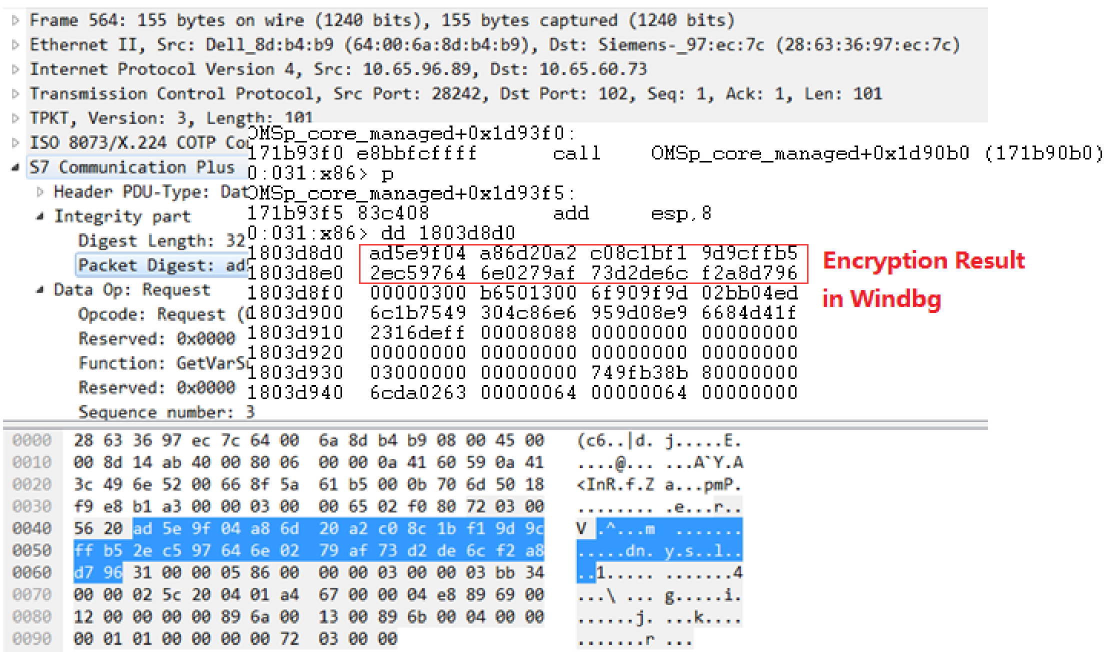


Figure 6.5 Function Encryption part in S7CommPlus Function packet

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

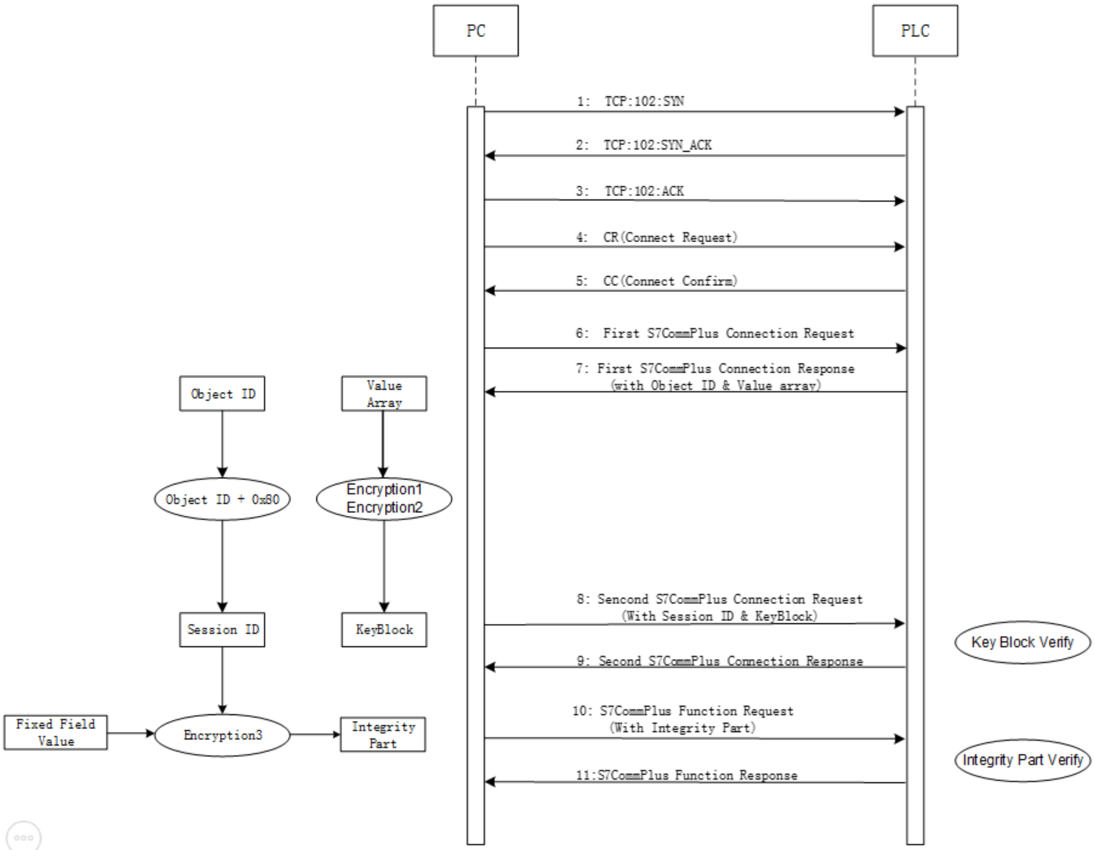
6.3 S7CommPlus Communication

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/.