PLC-导火线：工控 PLC 蠕虫的实现

原文：PLC-Blaster: A Worm Living Solely in the PLC
作者：Ralf Spenneberg, Maik Brüggemann, Hendrik Schwartke
翻译：gjden

摘要：众说周知，工业生产过程是通过可编程逻辑控制器（PLC）来控制的。 现在市场上许多 PLC 都配置了以太网口并且可以用 IP 进行通信。 我们将以西门子 SIMATIC S7-1200 机器为例展示一个蠕虫范例。此蠕虫不需要依赖 PC 电脑去扩散。该蠕虫仅仅活跃并运行在 PLC 当中。它可以通过网络扫描来寻找新的目标（新的 PLC），然后攻击这些目标并将自身拷贝到这些新的 PLC 中，而且目标 PLC 上运行原主程序不会发生任何改变。这些 PLC 目标一旦感染该蠕虫后会再次进行扫描感染。我们将分析蠕虫对目标的影响以及提出可能的缓解技术。

1. 介绍

IT 系统在当今工业生产发展至关重要的一部分。 可以说，没有现代化的通信网络，就不可能有现代化的工业生产。然而不幸的是，正由于工业系统对当今 IT 系统和通信网络的依赖，使得用户也暴露于被攻击的危险之中，而这早已是 IT 界久知的问题。IT 黑客攻击可能对工业系统造成多种伤害。比如他们可以造成工业生产的中断以及高额的经济损失，与此同时还可能对生活环境以及生命健康造成负面影响。其攻击的威力在 STUXNET 蠕虫上得到了充分的展示。西门子的可编程控制器(PLC)受到恶意篡改以阻碍伊朗核燃料铀浓缩。通过利用微软操作系统漏洞，该蠕虫感染进入到铀浓缩工厂的电脑当中，PLC 的软件被恶意篡改以摧毁铀浓缩离心机。该蠕虫需要一台 PC 电脑来进行传播并且通过 PC 电脑来攻击 PLC。这篇文章将将阐述一种可以在 PLC 之间传播的蠕虫。不需要任何 PC 电脑。蠕虫可能通过一个已经被感染过的 PLC 而被引入到工业工厂中，蠕虫接下来便会通过自我复制来感染到其他 PLC 中，并且在修改目标 PLC 并执行感染时添加到 PLC 用户程序中…。这篇文章描述的蠕虫是基于西门子 SIMATIC S7-1200v3，该蠕虫通过结构化文本（ST）编写而成，该语言是一种用于开发 PLC 软件的编程语言。

2. 相关工作

2015 年在美国 BlackHat 大会上，Klick 公司展示了一款在 PLC 上运行的恶意软件。他们使用 PLC 的一个通信特征实现了代理服务。我们将用同样的通信特性来实现了一个可以传送一个蠕虫程序的协议。通过使用该协议，该蠕虫可以从一个 PLC 直接传到另一个 PLC 中。这个蠕虫不需要更多系统去支持。我们没有用更加闻名的 SIMATIC S7-300，取而代之的是，我们的工作成果基于新的 S7-1200v3。PLC 使用的协议也与旧版不同。该文章也将会介绍这种新型的协议。

3. PLC 体系结构

PLC 使用简单系统结构构建而成。他们基于中央处理器（CPU）模块，加上数字输入和输出的模块组成。CPU 处理 PLC 操作系统以及运行用户程序。此外 CPU 还负责与其他设备通讯以及管理过程图（Process image）。

过程图（Process image）储存着所有的输入和输出的状态。用户程序不能直接操作物理的输入输出，而是通过操作过程图像而实现的。用户程序的运行是一个循环。每次循环的起始和结束的时候，CPU 会刷新过程图像。循环上限指的是循环时间。如果（一个循环运行的时间）超过循环上限时，PLC 会停止用户程序运行并抛出一个异常。

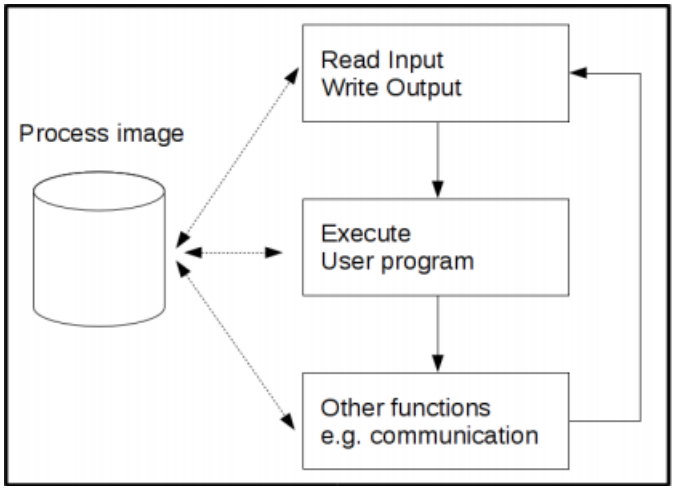

Figure 1. Zyklus eines PLCs

用户程序通过POU（程序组织单元）构造而成。这些单元（程序组织单元）包含了控制PLC的使用说明，因而可控制工业生产。SIMATIC S7-1200支持以下的POU：

- Organisation block (OB): 用户程序主入口
- Data block (DB): 全局存储器
- Function (FC): 功能
- Function block (FB): 有稳固局部储存器的功能

此外有几个西门子提供的给用户自定义的POU功能也可找到。这篇文章运用了系统 POU TCON和TDISCON。 使用这些POU，PLC可以初始化或者销毁任意系统的TCP连接。也可以通过TRCV和TSEND来接收和发送缓冲区数据。

4. 电脑蠕虫

电脑蠕虫从1988年就已经出现并且是恶意软件中有名的一种。蠕虫多种但都基本架构相同。所有的蠕虫攻击都可以归为一下几个阶段：可能目标的搜索，感染目标，在目标上执行，添加恶意功能。在PLC上蠕虫也同样支持这些工功能。这篇文章将展示每个必须组件的实现方法。

5. 在 S7 1200 上实现一个蠕虫

5.1 体系结构

这个蠕虫的编写与其他限制性蠕虫软件一样。在开发过程中，必须满足特定的PLC的限制条件，尤其是循环上限。蠕虫每几微秒就必须中断其执行，并在接下来的每次循环中能够继续执行。为了满足这个要求，我们通过状态机来设计该蠕虫。当前的状态被存在全局变量里面，在每次循环开始开始时，蠕虫的相关代码将会被调用。所以永远不会超过循环上限的时间。

如图2中所展示的步骤，蠕虫首先初始化一个连接来连接可能目标。一旦建立了连接，蠕虫会检测目标是否已经被感染，如果没有感染，蠕虫将暂停目标机上的用户主程序的执行，使其能够传输自己的代码。用户程序暂停后蠕虫将自己复制到目标PLC上，完成后启动目标PLC，接下来蠕虫再测试下一个可能的目标。

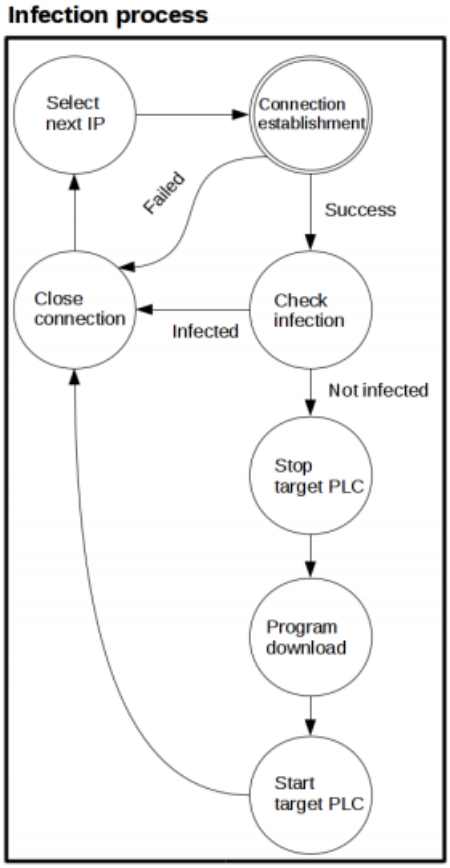

Figure 2. Execution sequence of the worm

5.2 目标侦查

蠕虫先扫描可能的目标，西门子的PLC可以通过102/tcp端口来识别。这个端口仅仅能够被外部防火墙关闭，并且其他普通的服务不会使用这个端口。

S7-1200用POU TCON管理TCP连接。这个POU的的用法在图3的第3行进行了展示。任意一个IP地址和端口在第9行代码中被传递。一旦POU被调用，PLC会试图建立连接，并且这两者是异步发生的。在之后的循环当中会对当前的连接状态进行验证。返回值DONE(Zeile 5)信号表示该连接是否被建立，如果为true，继续感染。如果IP地址和端口无法找到是不会有错误提示的。蠕虫需要在每次循环中通过递增计数器来检测超时。

```
1 IF "data".con_state = 10 THEN
2
3 "TCON_DB"(REQ:="data".action,
4 ID:=1,
5 DONE=>"data".con_done,
6 BUSY=>"data".con_busy,
```

```
7 ERROR=>"data".con_error,
8 STATUS=>"data".con_status,
9 CONNECT:="data".con_param);
10
11 IF "data".con_done = True THEN
12 // connection open
13 "data".con_state := 20;
14 ELSE
15 // connection not open
16 "data".con_timeout := "data".con_timeout + 1;
17 // connection timeout?
18 IF "data".con_timeout > 200 THEN
19 "data".con_state := 0;
20 END_IF;
21 END_IF;
22
23 GOTO CYCLE_END;
24 END_IF;
```

Figure 3. Target Detection using SCL

如果在200个循环后没有建立连接，蠕虫会继续执行在图4当中的代码。最然没有建立任何连接，但为了下一次连接的建立，必须调用POU TDISCON来释放资源。在第13行中，IP地址是递增的，因而所有的/24子网IP都会进行扫描并找到开放的102/tcp端口。

```
1 IF "data".con_state = 0 THEN
2
3 "TDISCON_DB"(REQ:="data".action,
4 ID:=1,
5 DONE=>"data".con_done,
6 BUSY=>"data".con_busy,
7 ERROR=>"data".con_error,
8 STATUS=>"data".con_status);
9
10 IF "data".con_error = True OR
11 "data".con_done = True
12 THEN
13 "data".con_param.REM_STADDR[4] := \
14 ("data".con_param.REM_STADDR[4] + 1) MOD 255;
15 "data".con_timeout := 0;
16 "data".con_state := 10;
17 END_IF;
18
19 GOTO CYCLE_END;
```

```
20 END_IF;
```

Figure 4. Target detection in SCL

5.3 感染

在感染阶段，蠕虫拷贝自身到目标PLC中。正常情况下软件通过西门子TIA-Portal（端口）传输到PLC中。蠕虫模仿TIA-Portal并实现了私有的Siemens协议。当我们在自己系统上分析这个协议时，发现Thomas Wiens的Wireshark plugin是可以解析这种协议的。

5.3.1 传输协议

文章的剩下部分主要讲解这种被称为S7CommPlus的私有协议。这是一个使用TPKT [6] 和ISO8073 [7]标准制定的一个二进制协议。正常情况下这两个协议都使用 102/tcp端口。

S7CommPlus主要特性：

- PLC的配置
- 启动和停止PLC
- 读写过程变量
- 程序传输（上传和下载）
- 调试
- 提供调试信息
- 警告

5.3.2. 消息

S7CommPlus所使用的每个消息都有着相似的结构。图5展示了连接中的第一个消息。TIA端口通过发送该消息来初始化一个连接。通用的结构接下来会进行解释。前两个域表示的是TPKT和ISO8073协议。他们的内容在相应的文档中都有解释。之后的0x72字节表示S7CommPlus信息的起始，不同的协议会有不同的版本号，长度域不受帧边界的限制。如果帧的边界丢失，更多的信息会附在附加的信息的后面。紧跟长度域后面的是类型域，子类型字段进一步指定了该消息。序列号是

随着每个消息增加而递增的。其他的数据是在特定属性块中传输的。

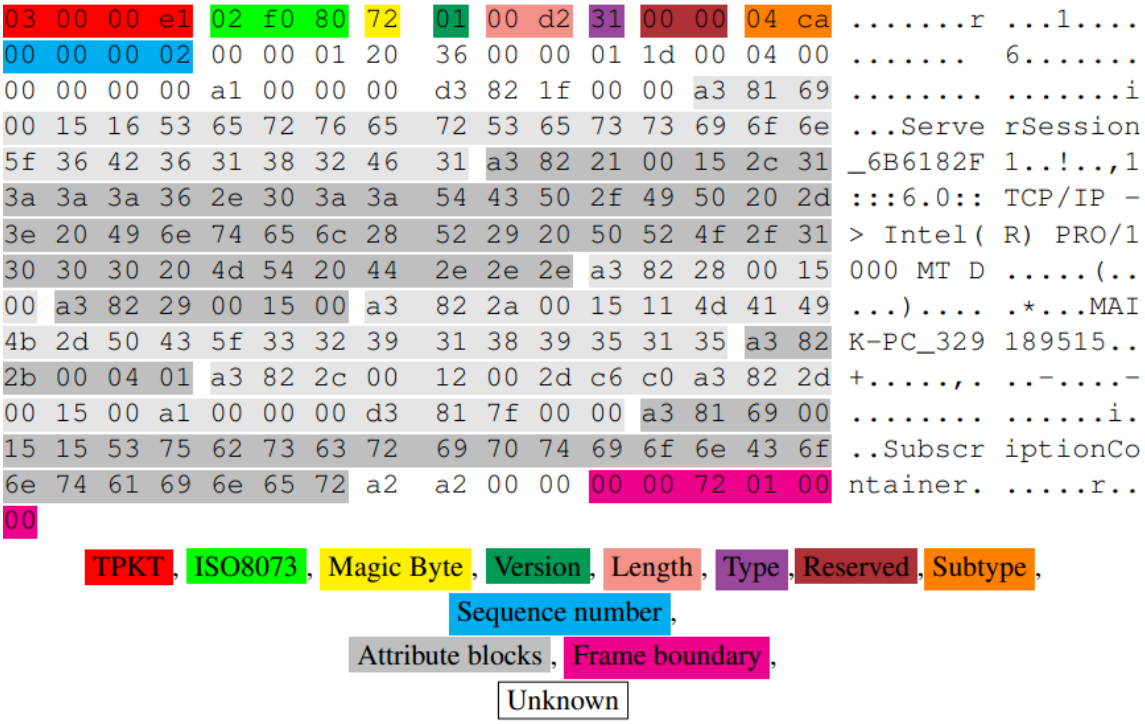

Figure 5. S7CommPlus message structure

5.3.3. 属性块

真正的数据是在属性块中。图6展示了上面例子中的第一个标志块。每个属性块都是以0xA3开始，该块包含一个字符串。完整的字符串包含长度和字符串值。

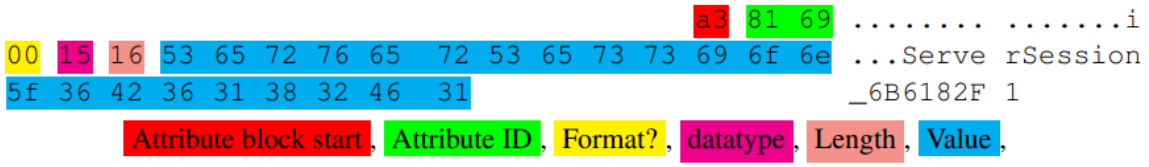

Figure 6. Attribute block

5.3.4. 数字编码

属性块中的数字通过一种特别的方式进行了编码。数字的长度是可变的，数字的每一个字节的 第一个位决定了之后是否还有字节数据。图7解释了上一个例子当中的属性ID和长度字段。此外，属性块中的值数据是不需要编码的。

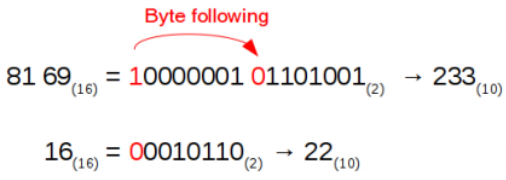

Figure 7. Encoding of numbers

5.3.5. 反回放机制

S7CommPlus协议可以检查到回放攻击。为了发现回放攻击，PLC所发送响应消息的第25个字节是一个随机数字，该字节数据用于检测回放攻击（图8）。随机数值在0x06和0x7f之间变化，这个字节称为anti-replay challenge。

TIA portal会基于该challeng数值做一次响应，响应的数据包通过第24、29个字节来指定检查值。检查值的计算公式如下：

antireplaybyte = challenge + 0x80

```
03 00 00 89 02 f0 80 72  01 00 7a 32 00 00 04 ca .......r ..z2....
00 00 00 02 36 11 02 87  22 87 3d a1 00 00 01 20 ....6... ".=.... 
82 1f 00 00 a3 81 69 00  15 00 a3 82 32 00 17 00 ......i. ....2...
00 01 3a 82 3b 00 04 82  00 82 3c 00 04 81 40 82 ..:.;... ..<...@.
3d 00 04 84 80 c0 40 82  3e 00 04 84 80 c0 40 82 =.....@. >.....@.
3f 00 15 1b 31 3b 36 45  53 37 20 32 31 32 2d 31 ?...1;6E S7 212-1
42 45 33 31 2d 30 58 42  30 20 3b 56 33 2e 30 82 BE31-0XB 0 ;V3.0.
40 00 15 05 32 3b 35 34  34 82 41 00 03 00 03 00 @...2;54 4.A.....
a2 00 00 00 00 72 01 00  00
```

Challenge, Attribute block

Figure 8. Anti replay mechanim

之后所有的从TIA端口发送给S7-1200的消息都需要在消息的第24位字使用anti-replay-byte。下图灰色的属性块部分也需要出现在这些消息中。

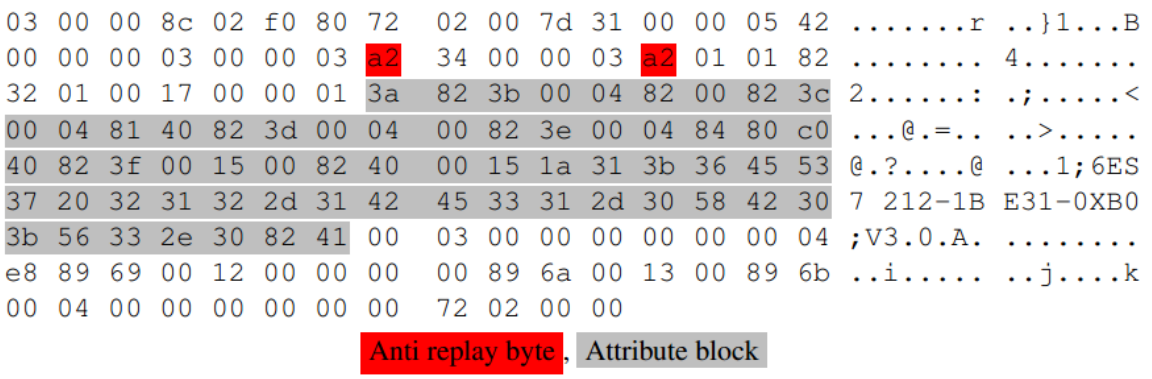

Figure 9. Anti replay Mechanism

5.3.6. 程序传输

为了传输用户程序，需要使用到一种特定的消息（图10）。每个消息传输一个POU。POU的类型也会因POU的不同而不同。块号（The block number）指定了该POU的PLC内存位置。

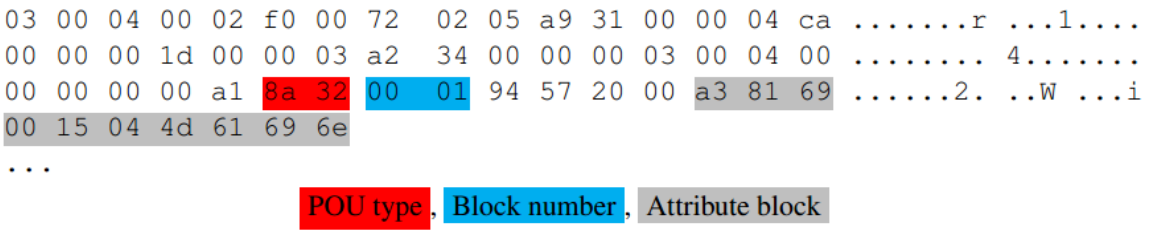

Figure 10. Transferring the user program

此外，消息头后面还会有几个属性块。在S7上存储有确切的字节码元信息，这个元信息详细说明需要的内存空间、创建日期、块号（block number）、所用语言、源代码和保护属性。TIA portal也许会使用这些信息来验证代码的有效性。

5.3.7. 确定所需的消息

在用户程序传输的过程中，有几个消息会进行交换，不过这个交换对于蠕虫来说并不是强制的。这些无关的消息会增加蠕虫的储存空间，因此被忽略。

图11展示了一次有效感染所需的消息。通讯首先被初始化，为了避免重复的感染，蠕虫首先测试目标并试图下载一个自己的拷贝。在上传代码之前，需要暂停PLC，然后传递程序，最后重启PLC。

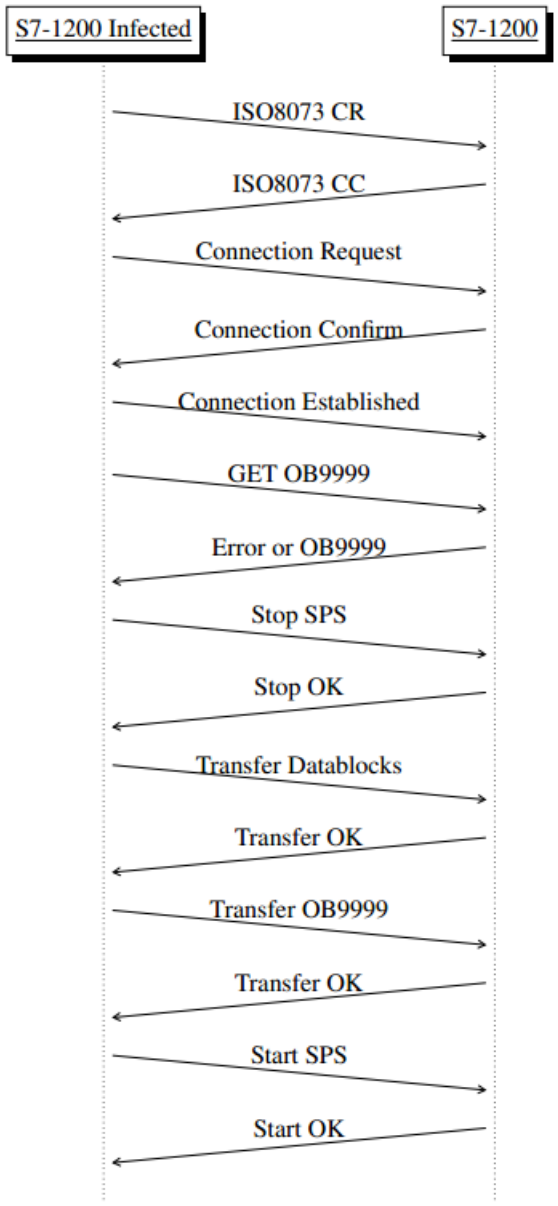

Figure 11. Messages exchanged during infection

5.3.8. 实现

基于以上协议的分析，传输程序可能会在PLC上被记录、修改和重放。到此，所有所需的消息都已知晓。为了在蠕虫中储存消息，需要使用到静态DB POU，额外的DBs用来储存临时变量和收发缓冲区。

静态DB必须储存感染所需所有消息。该DB块无法通过TIA portal来生成，需要手动进行构造。

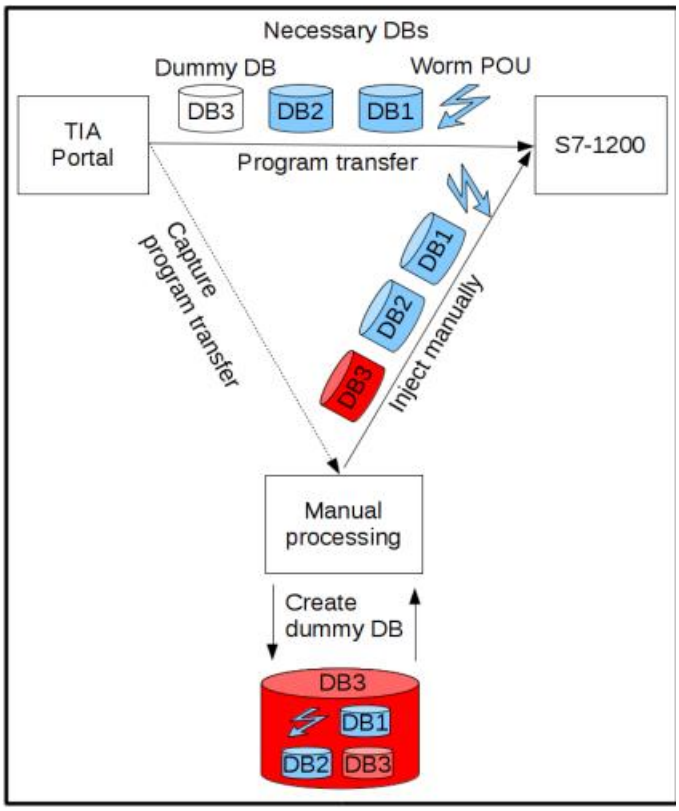

Figure 12. Manual generation of the DB

6. 启动蠕虫

当被传达的蠕虫代码加入到运行于目标PLC中的用户程序中时。额外OB和所需的DBs也需要加入。目标上的原始代码是不能够操作的，OB会被PLC自动发现并且执行。

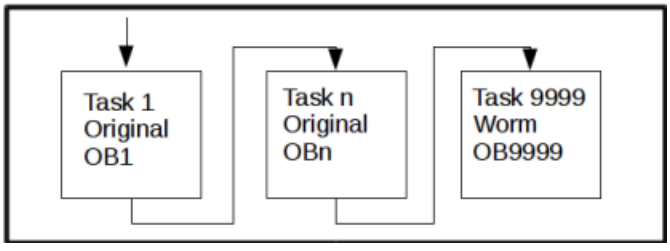

Figure 13. The worm is run as normal code

7. 恶意功能

根据蠕虫的功能特性，我们可以实现一些不同的蠕虫功能来阐述可能的危害性

7.1 C&C服务器

我们的蠕虫需要连接到C&C 服务器。通信协议是基于TCP的，蠕虫的各种功能可以通过C&C 服务器来触发。

7.2 Socks4代理服务

我们的蠕虫可以作为一个Socks4代理服务。一旦蠕虫连接上了C&C 服务器，在PLC网络中任何新的客户端的连接都通过的内嵌Sock4代理服务来启动。

7.3 DOS

PLC执行的时间超出循环上线，便会停止执行。蠕虫可以实现一个无线循环触发这样一种错误条件来达到Dos的效果。

7.4 操作输出

蠕虫可以操控PLC的任何输出，利用POU POKE，过程镜像中的任何值都可以进行篡改。

8. 蠕虫的探测、持久性与资源

8.1. 蠕虫的探测

8.1.1. TIA portal

TIA portal可以验证PLC上的用户程序，并且可以侦测到被篡改和被加入的POU（图14）。被红线框起来的是蠕虫所用的POU。因为TIA portal仅仅分析XML的源代码，所以对POU的分析是不可能的。此外通过利用TIA portal的bug中，蠕虫会导致应用崩溃。

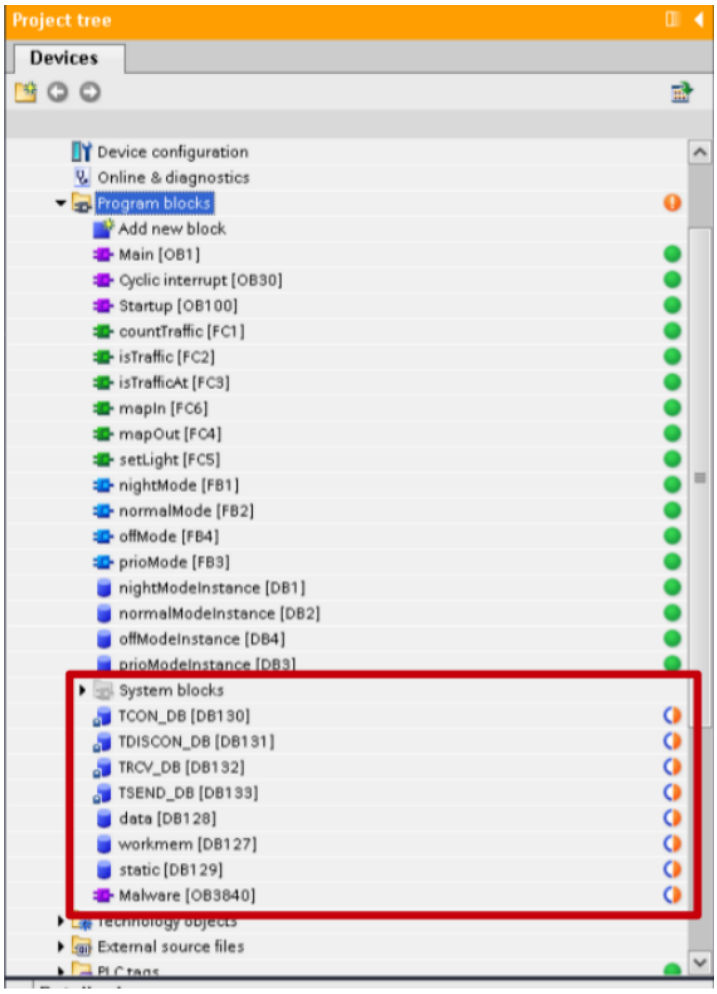

Figure 14. TIA portal exposes the worm

8.1.2. 停止PLC

PLC会在感染期间有大概10秒左右的时间是不工作的。在此期间原始的用户程序不会运行。这种PLC中断可能会引起注意并且被记录在PLC之中。

8.1.3. 网络流量

蠕虫会在ICS环境中产生不正常的流量。在扫面和感染阶段，会发送很多可疑的数据包。

8.2. 持久性

8.2.1. 重新启动与重新引导

蠕虫是被储存在PLC中的。它成了用户程序的一部分，所以即使在重启和甚至是拔除电源的情况下都会存在。

8.2.2. 恢复出厂设置

TIA端口可以触发PLC恢复出厂设置。这种情况下，所有的设置和用户程序包括蠕虫都会被清除。

8.2.3. 程序传输

我们的蠕虫存储在OB9999里面。如果这个POU被覆盖住了，蠕虫也就从PLC上被删除了。

8.3. 资源

8.3.1. 循环时间

循环上限是固定的，默认限制时间是150微秒。蠕虫一定不能超过这个限制。在没有用户程序的情况下我们测量的PLC循环时间是0微秒。在我们用蠕虫感染了S7后，再测量了一遍得到最大的循环时间是7微秒（ms），这是循环上限的4.7%。

8.3.2. 储存器

蠕虫需要38.5KB的RAM来存储蠕虫代码和数据。其中9.0KB（23.3%）是恶意功能代码。此外，还需要216.6KB的FLASH。表格1当中展示了不同型号机的储存器可用空间。

Model	RAM	Flash
S7-1211	50KB (77%)	1MB (21%)
S7-1212	75KB (51%)	1MB (21%)
S7-1214	100KB (38%)	4MB (5%)
S7-1215	125KB (30%)	4MB (5%)
S7-1217	150KB (25%)	4MB (5%)

Table 1. Memory footprint

9. 保护机制

PLC S7-1200v3提供三种保护机制。我们会依次分析这些机制并确定这些机制是否能够保护PLC免收蠕虫感染。这个分析报告是基于TIA portal V11 SP2 Update 5 和使用firmware 3.0.2的S7-1200

9.1 专有技术保护（Knowhow Protection）

专有保护技术保护用户程序不被未授权访问。用密码可以禁止非授权访问和POU篡改。

专有保护技术通过属性块儿实现的，该块在程序传输的时候被写入到PLC中。在图15中展示了该块。Flag表示该技术是否启用。password hash 是基于密码P使用如下公式产生的：

H = sha-1(encode_utf-16le(P))

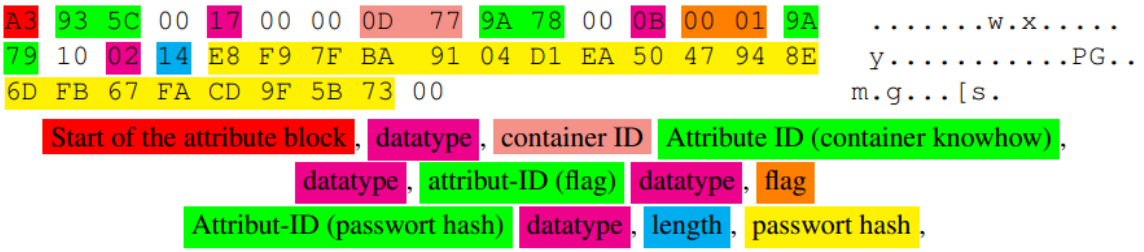

Figure 15. Attribute block: knowhow protection

TIA端口获取该属性块。如果flag被设置了，TIA portal在密码不正确的情况下会禁止读和写相关程序块。密码是通过hash进行比对的。

为了避免代码被访问，PLC上XML的源代码是通过AES128-CBC加密的。图16展示了加密的源代码。

```
<NetworkContainer>
 <Network Lang="SCL" ProgrammingContext="Plain"
        Mnemonic="International" RefID="1">
  <EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Element"
          xmlns="http://www.w3.org/2001/04/xmlenc#">
   <EncryptionMethod
           Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc" />
   <CipherData>
    <CipherValue>
         AQIDAQIDAQIDAQIDAQIDAS/acA/s+L3CoQYaeAQMOq ...
    </CipherValue>
   </CipherData>
  </EncryptedData>
...
```

Figure 16. Encrypted source code

9.1.1. 薄弱点

完整性保护缺失

即使在专有保护技术的存在下，块还是有可能被读取和修改。该保护特征是在TIA portal中实现的，而不是在PLC里面。使用自写工具在PLC中进行块的读写是可以的，PLC中拥有绝对的权限。甚至能够直接重置专用保护技术flag，以至于可以通过TIA portal来实现完全访问。

AES-key可能被衍生

AES加密key可能来自于 password hash。 password hash可以通过自写入软件来读取。Key是采用如下公式计算得到的：

```
K = truncate128Bit(H) XOR M
```

with M :

```
M = 0x28,0x6f,0x76,0x5c,0x6e,0x3b,0x1e,0x4c,
    0xd0,0x8e,0x42,0x31,0x43,0x7b,0x8e,0xbf
```

Fixed by the vendor in SSA-833048[1].

.

9.1.2. 从蠕虫的角度分析保护功能

该特性不能够防止蠕虫攻击

9.2. 防拷贝保护

防拷贝保护禁止将用户程序拷贝到另外一个PLC中。目标PLC的序列号是存在于用户程序中，以此防治TIA Portal将用户程序传输到其他PLC中。序列号被储存在几个独立的属性块当中。

9.2.1. 更多薄弱点

完整性保护缺失(供应商在SSA-833048中已经修复)指的是属性块的完整性未经任何保护。存储其中的序列号可能被修改甚至删除。PLC本身并不会检查序列号，这种保护特性仅仅是在TIA portal中实现的。

9.2.2. 蠕虫的角度来分析防拷贝保护机制

该（防拷贝特性）特性也不能防止该蠕虫的攻击

9.3. 访问保护

访问保护可以防止用S7CommPlus协议软件实现无密码访问PLC。其有三种保护级别，表格2列出了不同的级别。

Function	None	Write	Read/Write
Start/Stop CPU	yes	no	no
Write Program	yes	no	no
Read Program	yes	yes	no
Manipulate memory/output	yes	yes	yes
Read identification	yes	yes	yes
Setting IP address	yes	yes	yes
Setting date	yes	no	no
Factory reset	yes	no	no

Table 2. Access based on the protection level

这些级别的验证采用的是challenge响应机制【8】。

9.3.1. 从蠕虫的角度分析访问保护机制

该访问保护机制可以保护PLC免受蠕虫的攻击。写保护可以防止任何人篡改PLC中的代码。使用challenge响应机制来鉴定应该说是安全的。如果蠕虫不知道密码，那么就无法感染PLC。但是在默认情况下访问保护都是关闭的。

10. 结论

在这篇文章中我们阐述了PLC蠕虫实现的可行性。该蠕虫代表了工控行业中的一种新型威胁。这样的网络在传统观念里都很好地受到保护而难以受外部攻击。通过引入PLC蠕虫，PLC成为了攻击源，而不只是攻击目标。受感染的PLC可能通过工控组件提供商或者在组件传输过程中被掺入其中而进入到工控系统中。蠕虫之后便可以在工控网络内部进行扩散，且不需要任何标准电脑和服务器。因此它不会被任何杀毒软件做侦测到。此外工厂操作员也只有很少选择来识别PLC上的恶意软件。

参考：

[1] Eric Chien Nicolas Falliere, Liam O Murchu. W32.Stuxnet Dossier. Last accessed

https://www.symantec.com/content/en/us/enterprise/media/

security_response/whitepapers/w32_stuxnet_dossier.pdf on

06.02.2016, 2011.

[2] Johannes Klick u.a. Internet-facing PLCs - A New Back Orifice.

https://www.blackhat.com/docs/us-15/materials/

us-15-Klick-Internet-Facing-PLCs-A-New-Back-Orifice-wp.

pdf, 2015.

[3] pressetext.deutschland. Malware-Jubiläum: 20 Jahre Internet-Würmer.http://www.

pressetext.com/news/20081101001, 2008.

[4] Nicholas Weaver u.a. A Taxonomy of Computer Worms. https://www1.icsi.

berkeley.edu/~nweaver/papers/2003-taxonomy.pdf, 2003.

[5] thomas_v2. S7comm Wireshark dissector plugin. http://sourceforge.net/

projects/s7commwireshark/files/.

[6] Dwight E. Cass Marshall T. Rose. ISO Transport Service on top of the TCP. Last accessed

https://tools.ietf.org/html/rfc1006on 21.02.2016, 1987.

[7] International Organization for Standardization. Connection oriented transport protocol.

ISO Std. 8073, 1988.

[8] SCADAStrangeLove. S4x13 Releases: S7 password offline bruteforce tool. http:

//scadastrangelove.blogspot.de/2013/01/s7brut.html, 2013.