

ROMHACK

2021

Saturday 25th of September

Making your own Stuxnet: Exploiting new vulnerabilities and voodooing PLCs

Nicolas Delhayé | [@_Homeostasie_](#)

Flavian Dola | [@_ceax](#)

AIRBUS
CyberSecurity

Who are we?

Nicolas Delhaye (@_Homeostasie_)

- Vulnerability researcher since 2010
- Specialized in Windows OS & applications
- Mainly rely on reverse engineering with a bit of fuzzing
- Recent findings
 - CVE-2020-26562 & CVE-2020-26871: DoS on Cybereason EDR
 - CVE-2020-7523: LPE in Schneider Electric
 - CVE-2020-1301 (SMBLost): SMBv1 RCE
 - CVE-2019-18568: LPE from kernel code execution in Avira Antivirus
 - CVE-2019-18567: Kernel DoS in Bromium

Flavian Dola (@ceax_)

- Vulnerability researcher since 2009
- Specialized in embedded systems: IoT, ICS, On Board components and so on
- Expertise on reverse engineering with a lot of fuzzing
- Recent works:
 - CVE-2020-7475, CVE-2020-28211, CVE-2020-28212, CVE-2020-28213: Various vulnerabilities on Schneider Modicon products
 - CVE-2020-24672: RCE on ABB SoftControl PLC Simulator
 - IP2LoRa: IP tunnelling over LoRA
 - afl_ghidra_emu: Fuzz exotic architecture using AFL & Ghidra emulation engine

*Thanks to **Guillaume Orlando (@HomardBoy)**: exploit development & integration*

Agenda

- 1. Context**
- 2. Overview of the attack scenario**
- 3. Finding & Chaining vulnerabilities on the engineering station**
 1. Step 1: Accessing the engineering station (RCE)
 2. Step 2: Getting the NT AUTHORITY\SYSTEM rights (LPE)
 3. Step 3: Interacting with the PLC (intrinsic/hidden functions)
 4. Step 4: Designing and executing PLC's unconstrained code
- 4. Some key takeaways**
- 5. Q/A**

Context

Many critical companies own an ICS (Industrial Control System): energy, water, and so on

■ Risks

- Large attack surface: OT infrastructures rely on many software and many hardware components
- Compromising OT networks from an IT (Information Technology) access

■ Old well-known APTs

- **Stuxnet**: Sabotage Uranium enrichment
 - Siemens S7-300 and S7-400 PLC
 - Step7 engineering software
- **Triton**: Petrochemical plant
 - Schneider Triconex MP3008 – Safety controller
 - TriStation engineering software



Modicon 340 PLC



Overview of the attack scenario

Objective: Compromise PLCs from the IT network

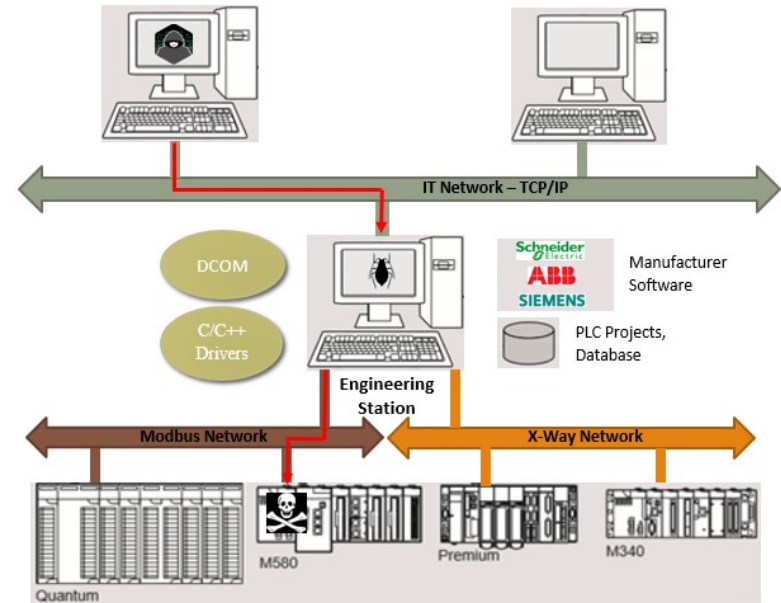
Target: The Engineering Station (ES)

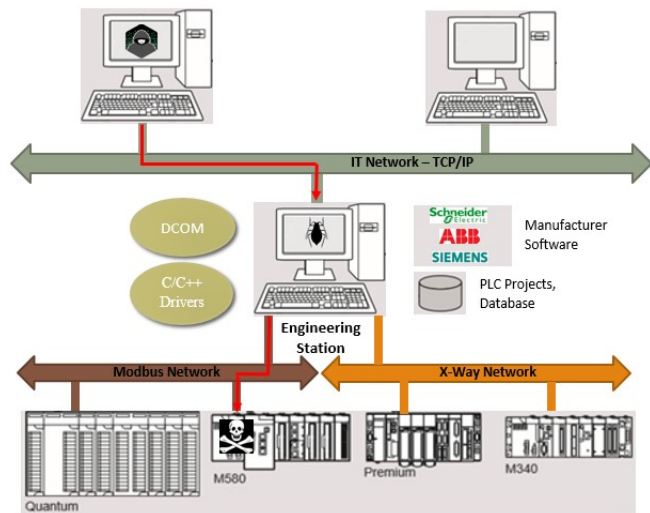
Why?

- Relevant target to infiltrate in-depth OT
- A legitimate station to exchange with PLCs
→ Load and update programs
- Far too often connected to OT

What we need?

1. RCE to access the target
2. LPE to gain full control of the target
3. PLC's unconstrained code execution



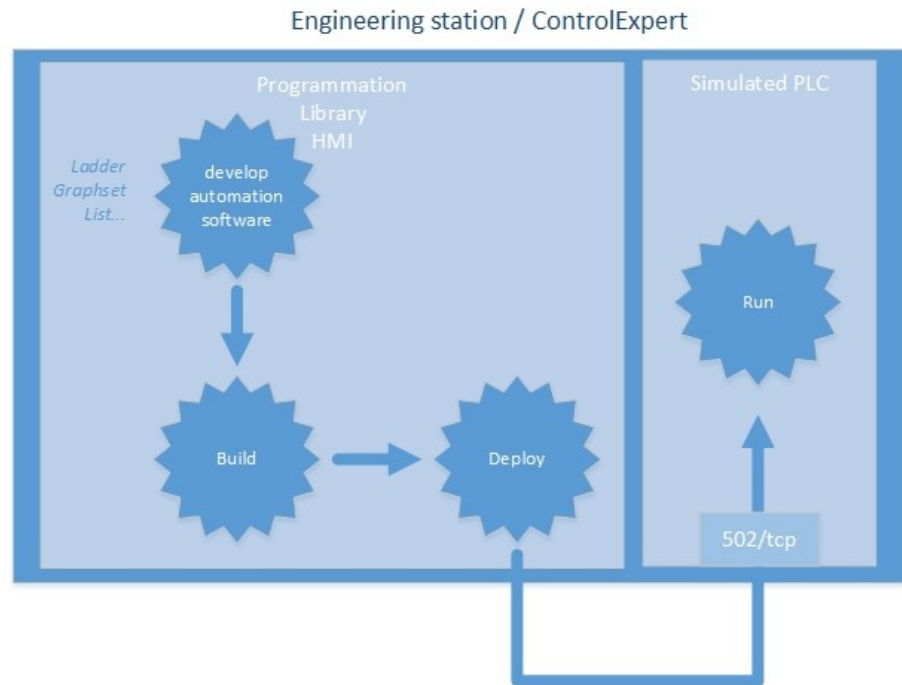


Step 1: Accessing the engineering station

CHAINING 3 VULNERABILITIES TO GET A RCE
(CVE-2020-28211, CVE-2020-28212, CVE-2020-28213)

Target: PLC simulator

- Part of the engineering software (ControlExpert)
 - ➔ Used when software developer wants to test his automation program
 - ➔ Implement the same protocols that real one (Modbus, UMAS)
- 502/tcp port opens on all network interfaces! Attacker can deploy its own program
 - ➔ Escape sandbox to take control of the engineering station?
 - ➔ Closer look of how automation program is deployed

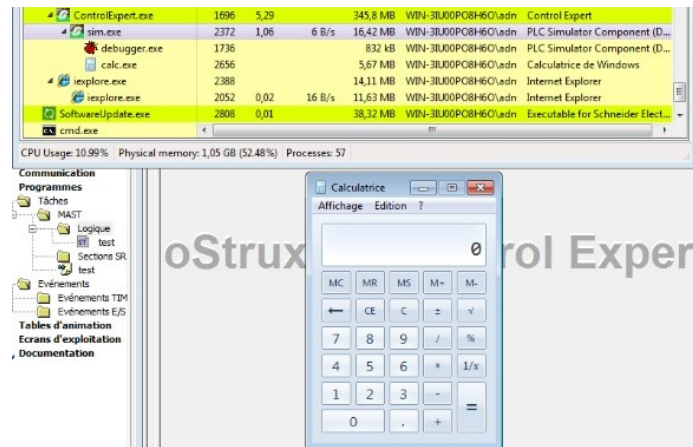


Vuln 1: Unconstrained RCE CVE-2020-28213

- **codeengine.dll** is in charge of compile automation program
- Many arch are supported:
 - ✓ ARM
 - ✓ Sonix ???
 - ✓ Intel32
- Native machine code is generated!
 - ✓ Break after compilation
 - ✓ Replace with your shellcode
 - ✓ Deploy on sim PLC... Tadaa...

□ No sandboxing mechanism

```
.rdata:100BD524      const CVirtualEngine::'vftable' dd offset sub_100340D0
.rdata:101047F4      const CArmEngine::'vftable' dd offset sub_10045280
.rdata:1015545C      const CIntel32Engine::'vftable' dd offset sub_10053340
.rdata:101E7F6C      const CSonixEngine::'vftable' dd offset sub_1005D570
.rdata:10205724      const CLangcEngine::'vftable' dd offset sub_1006C710
.rdata:10230214      const CLangcArmEngine::'vftable' dd offset sub_10085010
.rdata:10231934      const CLangcIntelEngine::'vftable' dd offset sub_1008FC60
```



Vuln 2: Hijacking UMAS session CVE-2020-28212

- PLC simulator accepts 1 connection at a time
- Attacker must find a way to disconnect legitimate user connection
- Connection is based on UMAS protocol
 - ✓ Very old proprietary protocol on top of Modbus layer
 - ✓ Lack of authentication and ciphering
 - ✓ No correlation with TCP session
- Vulnerable to man on the side attack

```

□ 5a 00 10 ... REQ_TAKE_RESERVATION
□ 5a 00 fe 78
□ 5a 78 11     REQ_RELEASE_RESERVATION
  
```

- Session ID is only encoded on 1 byte
- Brute force takes just few seconds!

```

_work_with_ReleaseReservation proc near

pMystreamReq= dword ptr 8
pMystreamResp= dword ptr 0Ch
pRes= dword ptr 10h
pUmasPayloadResp= dword ptr 14h

000 push    ebp
004 mov     ebp, esp
004 mov     cl, g_resa_id_
004 test    cl, cl
004 jz      short loc_10001C34

004 mov     eax, [ebp+pMystreamReq]
004 cmp     cl, [eax+my_stream.t.resaId]
004 jnz     short l_err
  
```

Vuln 3a: By-pass project authentication CVE-2020-28211

- ControlExpert recommends to start the Simulated PLC with protected project inside
 - Do not know password □ Can not load a new project in simulated
- Protection originally designed to protect intellectual properties of automation projects
- Network capture during authentication process
 - No network traffic on password check
 - Authentication is done on client side
 - What happens when client side is controlled by an attacker...
 - Just have to send the right function code “Authentication succeeded” to gain access!

Vuln 3b: By-pass project authentication CVE-2020-28211

- Going deeper...

```
lea    eax, [ebp+v_res]
push   eax                ; pRes
mov    ecx, [ebp+var_348] ; void *
call   asrhmiverifyapplicationpwd

movzx  ecx, [ebp+v_res]
test   ecx, ecx
jnz    loc_1007D7FD
```

- Patch “verify password” function on ControlExpert to return “Success”
 - No ciphering mechanism based on password provided!
 - **You can access to the automation source code despite of the project protection**



- ❑ Considering attacker had access to SI network

Demo – Engineering Station infection

RCE on the engineering station

1. Build malicious simu project
2. Disconnect existing UMAS sessions
3. ByPass simu auth
4. Load malicious project in engineering station simuPLC
5. Execute malicious project
6. Payload connect back to C&C

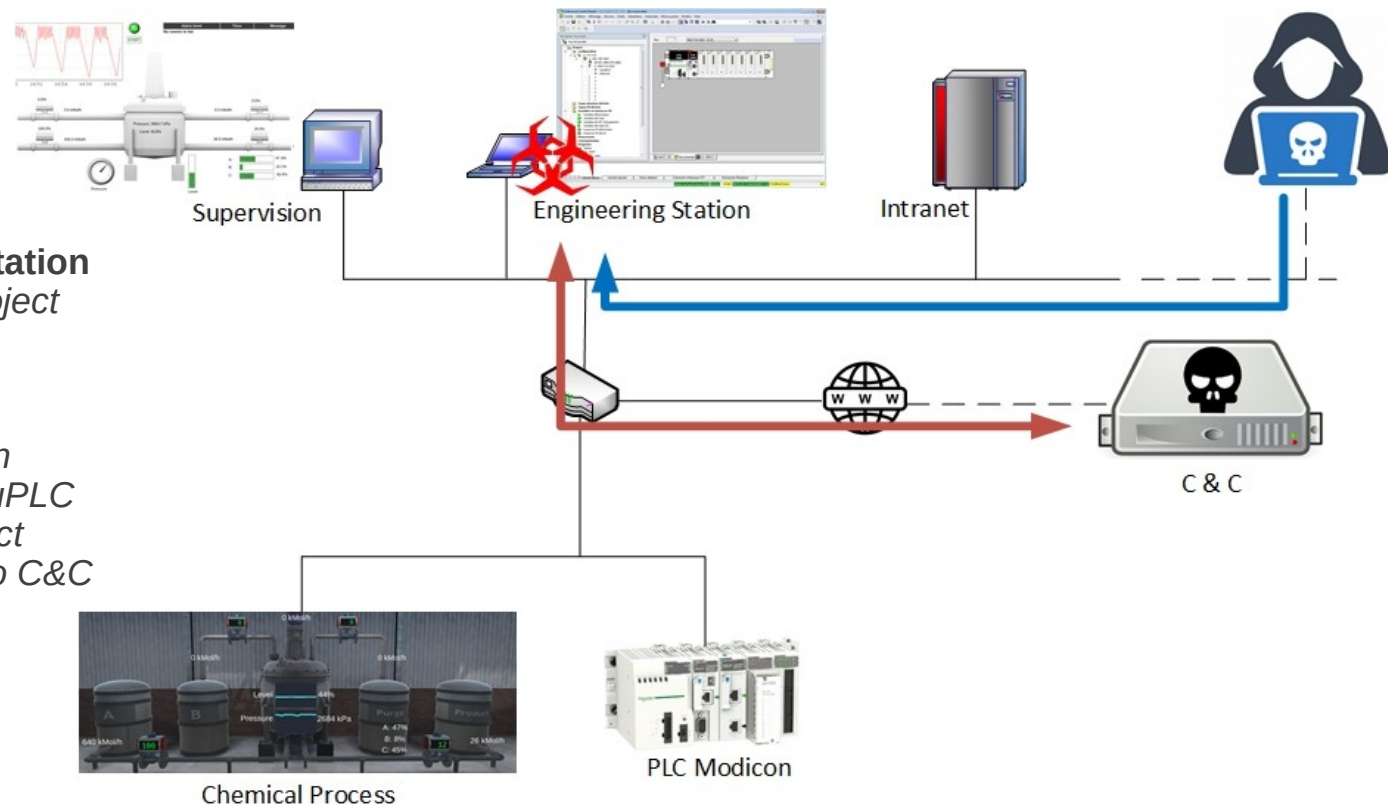



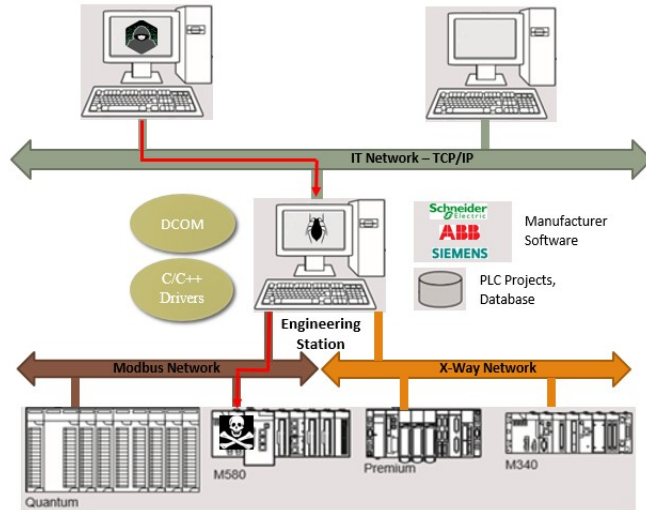


ABB RCE on PLC Simulator CVE-2020-24672

- We found the same vulnerability class in another vendor: **ABB SoftController** (CVE-2020-24672)
- ABB Simulator PLC is listening on all network interfaces `0.0.0.0:102/tcp`
- ***ControlBuilderStd.exe*** transforms automation program to x86 native code

 CodeGeneratorIntel::WriteBinaryCode(uchar * *)	.text	00BB9EC0
 CodeGeneratorPPC860::WriteBinaryCode(uchar * *)	.text	00BCD0C0
 CodeGen_v50::CodeGeneratorPPC860::WriteBinaryCode(uchar * *)	.text	00BDF A10

- By patching generated machine code you can execute your own code in the remote target:
 - At least RCE in simulated PLC
 - Maybe also in physical PLC (not tested on ABB PLC)
- **Design vulnerability class: Certainly presents in other ICS manufacturers!**

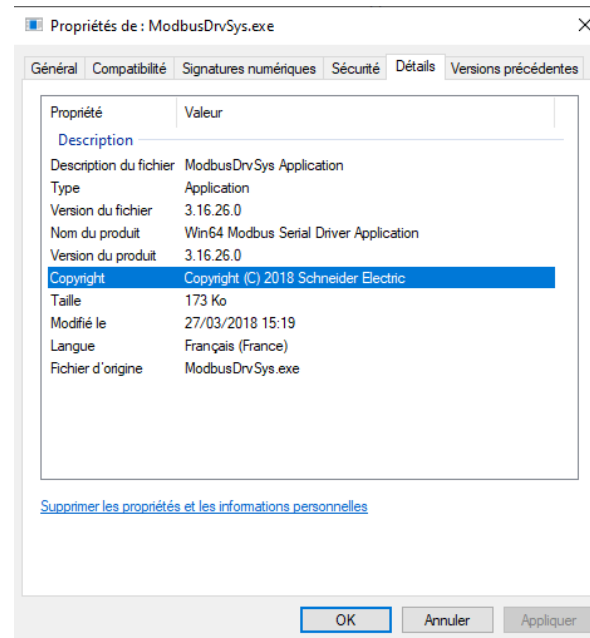


Step 2: Getting SYSTEM rights

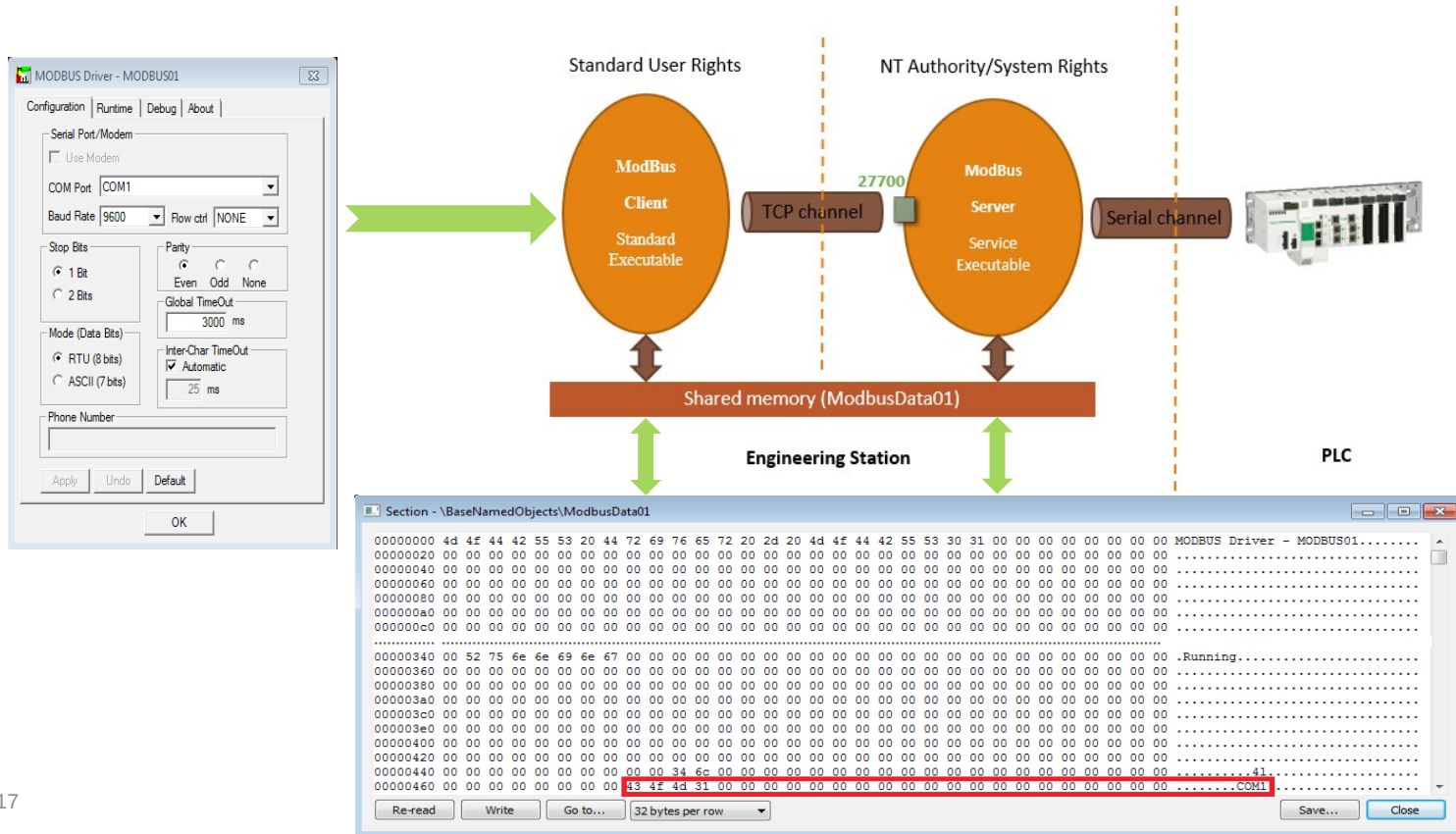
LOCAL PRIVILEGE ESCALATION (CVE-2020-7523)

Overview

- **Schneider environments with Ecostruxure Control Expert** software (formerly known as **Unity Pro**)
- Bug located into a critical component
 - Named **ModbusDrvSys.exe**
 - Service running *Authority NT/System* rights
 - Perform the Modbus communication
- Not straightforward to exploit
 - Properly abuse the shared memory
 - Find a way to control data to overwrite the file
 - Bypass one restriction related to the length of the filename
 - Locate a relevant executable to overwrite it



Basic architecture of Modbus Communication



CVE-2020-7523: LPE Exploitation

Limitations

1. “ModbusDrvSys.exe” service only allows you to open a 13 bytes-length filename



2. Dealing with a maximum of 1042 bytes to overwrite the targeted file



3. Finding a targeted executable running with NT/Authority rights



Counter-measures

1. Thinking about NTFS junctions and object manager symbolic links
□ “An introduction to privileged file operation abuse on Windows” by @clavoillotte

2. Building a tiny PE (based on pts-tinytype GitHub project)

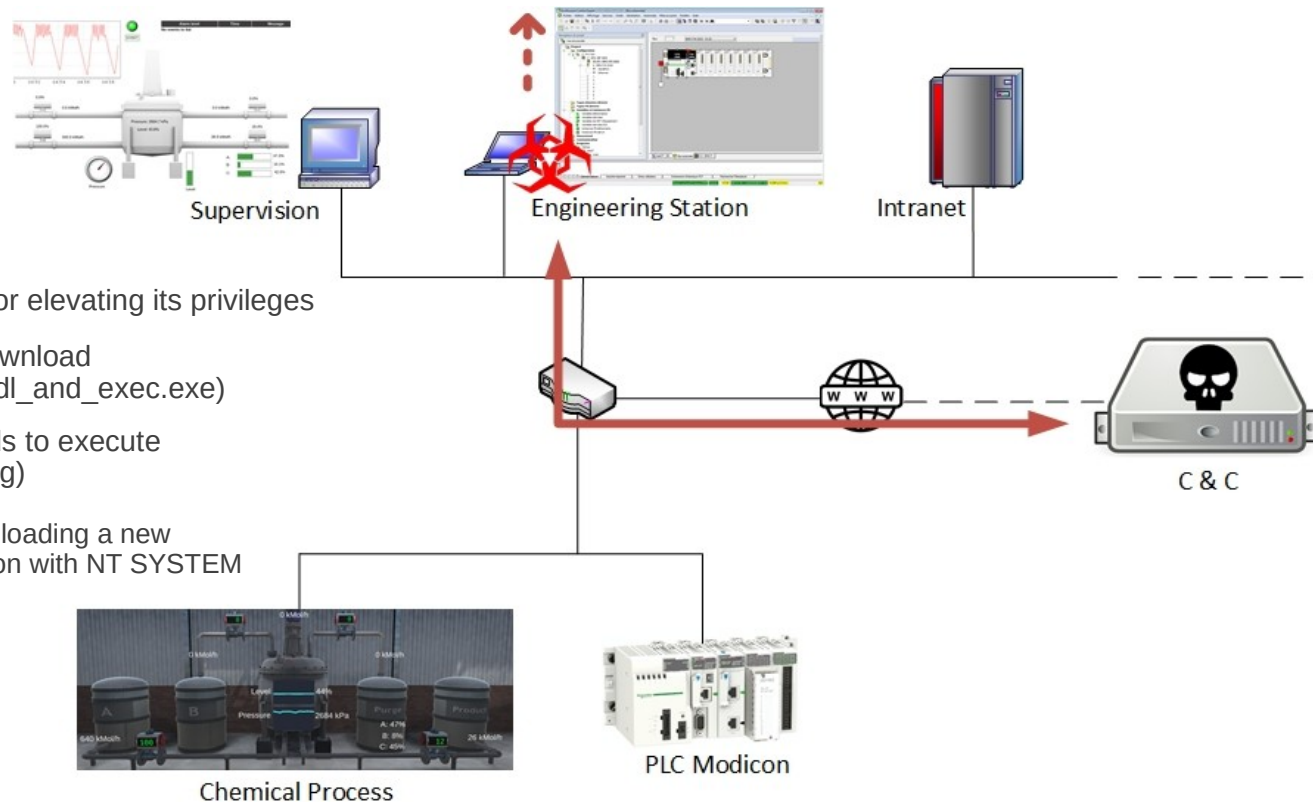
□ The smallest “hello world” PE (402 bytes) runs from Windows XP to Windows 10! □

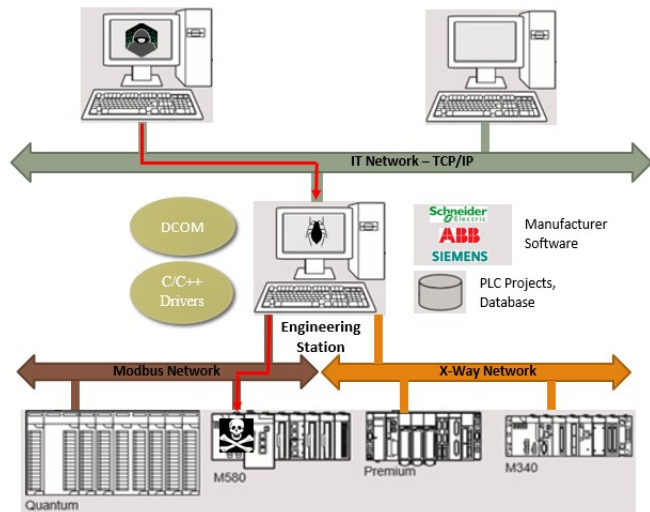
3. A Schneider Electric executable named “NA_MBP.exe” (NetAccess Application)

Context:

Malware executed from the RCE on the PLC simulator requires SYSTEM rights

1. Attacker or Malware requests for elevating its privileges
2. C&C informs it about files to download (CreateSymLink.exe, LPE.exe, dl_and_exec.exe)
3. C&C informs it about commands to execute (symlink, exploit, logoff, cleaning)
4. "dl_and_exec.exe" results in downloading a new malware establishing a new session with NT SYSTEM rights





Step 3: Interacting with PLCs

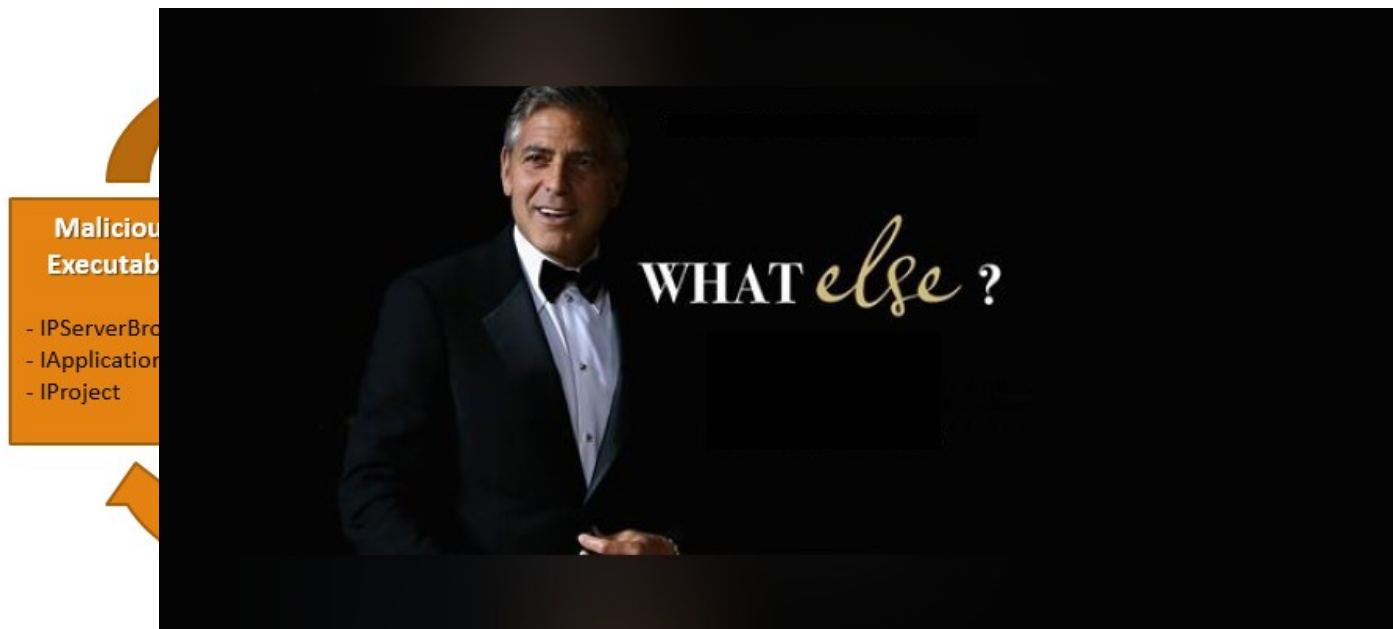
LOOKING FOR INTRINSIC OR HIDDEN FUNCTIONS

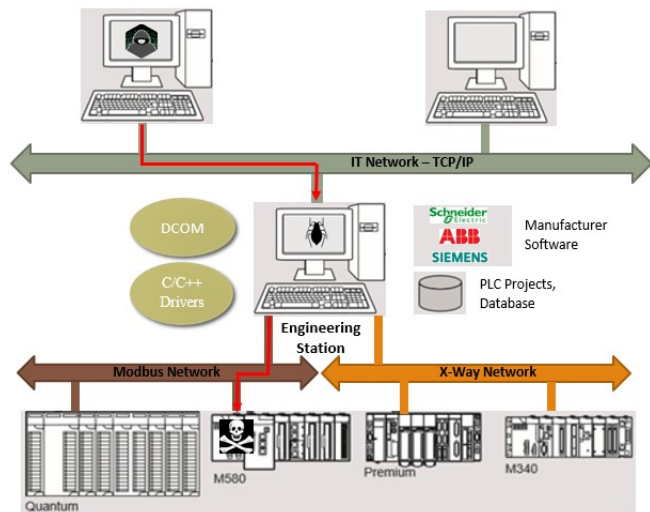
Legitimately infecting PLCs

- **Objective:** Communicate with PLCs to inject new payloads
 - ➔ Using the most reliable approach
 - ➔ Without user interaction
- **Outlines of PLC development projects**
 - ➔ Start a project
 - ➔ Launch “analyse” and “build” commands
 - ➔ Control PLCs by sending useful commands (start, stop, download...)
- **Attacker thoughts:** Reproducing this behaviour
 - ➔ Studying and relying on how the software works
 - ➔ Many thanks to DCOM technology with these reusable components



A case study based on DCOM





Step 4: Designing and executing PLC's unconstrained code

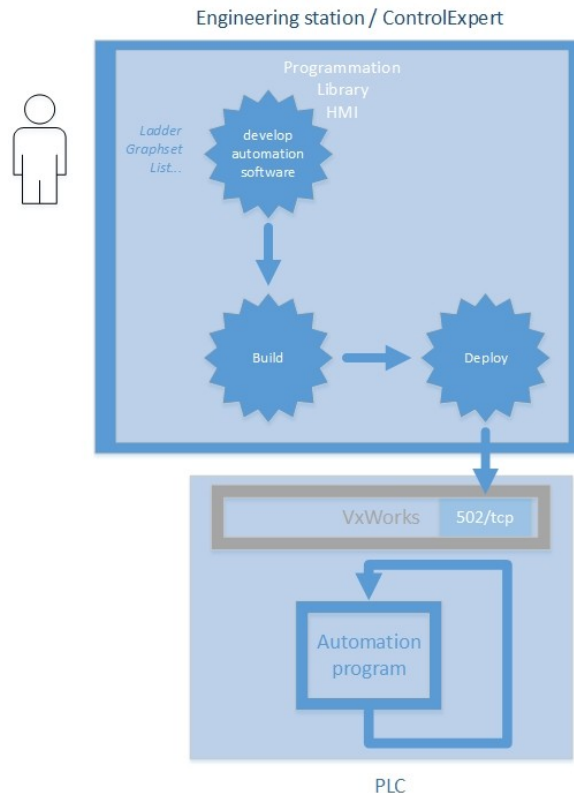
INJECT NATIVE CODE AND EXTEND PLC CAPABILITIES

(CVE-2020-7475)

PLC malicious code injection

CVE-2020-7475

- Same steps are used to deploy automation program on physical PLC
- ControlExpert generates ARM native code
 - by hooking compilation process, we can inject our own byte code!
- Need to inject our backdoor somewhere in automation program
 - Do not disturb physical process
 - Backdoor must be tiny and fast
 - Ability to take control on physical process if needed



Design PLC downloadExec backdoor

- C:/punitemp/: log ASM source code (Thanks Schneider!)

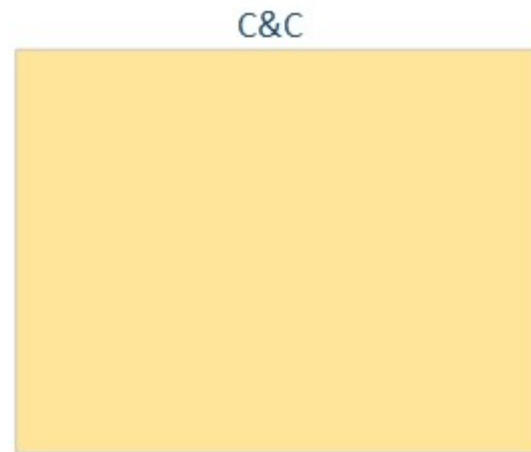
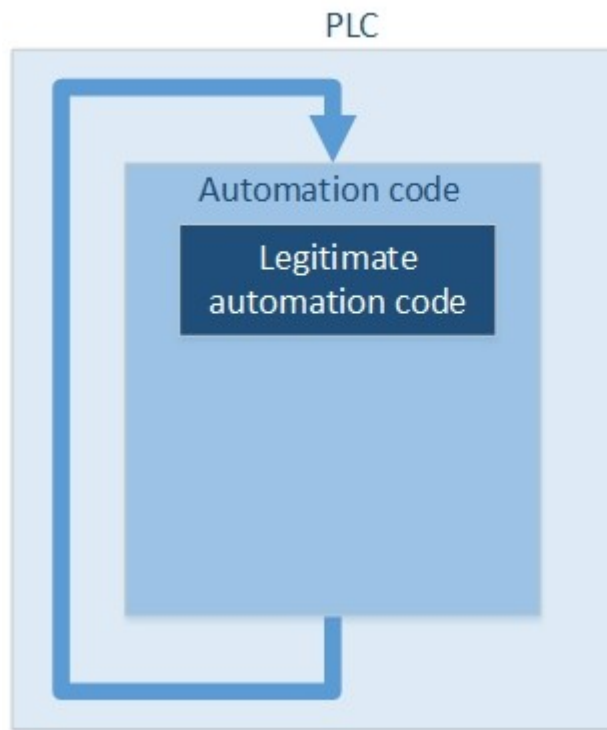
```
fc_pos_update_real := (fc_flow_set_real - fc_product_flow_real) * fc_flow_k;
```

LDR R0,[R7,#0xc] /* scaled_real */	MOV LR,PC
STR R0,[R8,#0x14] /* fc_flow_set_real */	MOV PC,R2
LDR R0,[R8,#0x0] /* fc_flow_k */	LDR R1,[SP],#4
STR R0,[SP,#-0x4]!	ldr r2,[pc,#0fffInPool14] from ldr r2,=mul_f
LDR R1,[R8,#0x10] /* fc_product_flow_real */	MOV LR,PC
LDR R0,[R8,#0x14] /* fc_flow_set_real */	MOV PC,R2
ldr r2,[pc,#0fffInPool13] from ldr r2,=sub_f	STR R0,[R8,#0x1c] /* fc_pos_update_real */

- First payload: “Dumper”
 - Dump OS (VxWorks), Schneider Electric functions, automation code
 - Reverse to locate OS functions (socket, malloc, ...)
 - Extend PLC functionalities ;-)

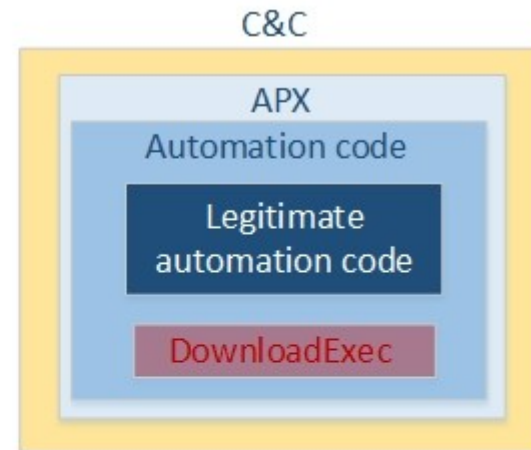
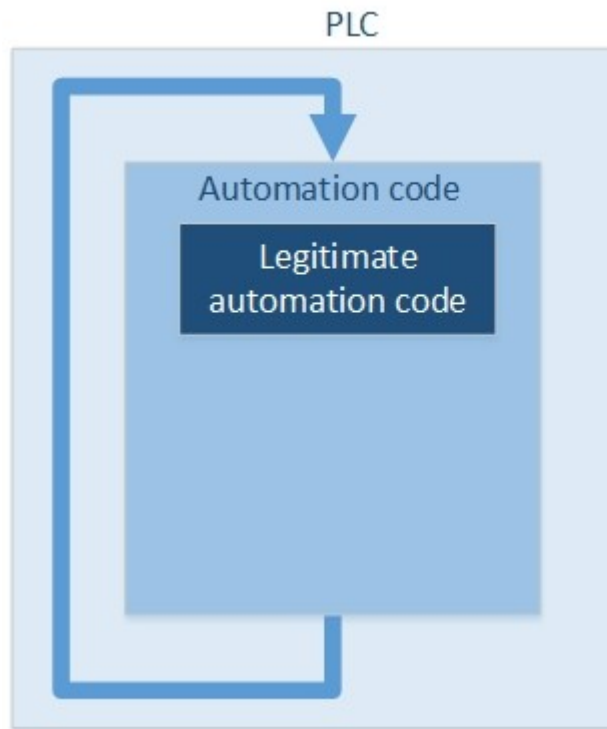
PLC backdooring

PLC is running with legitimate program inside



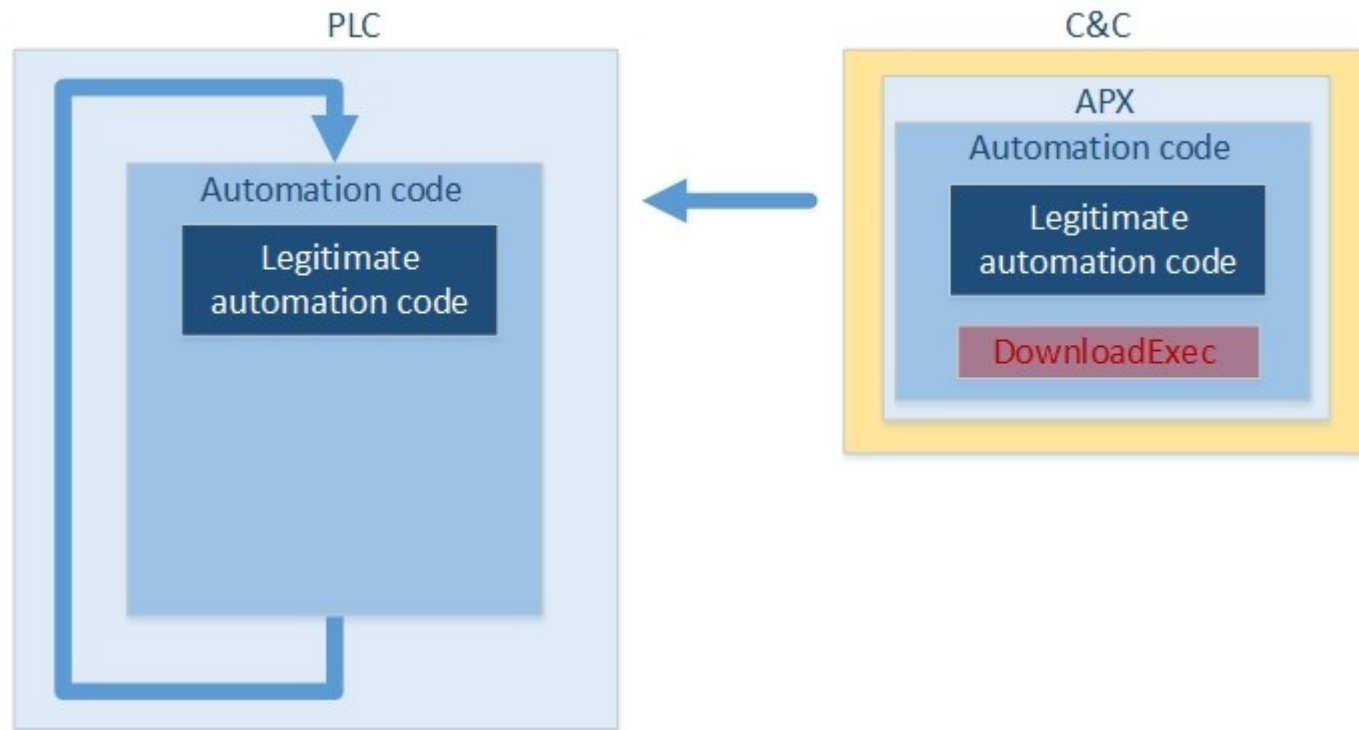
PLC backdooring

Attacker backdoors automation program with downloadExec payload inside



PLC backdooring

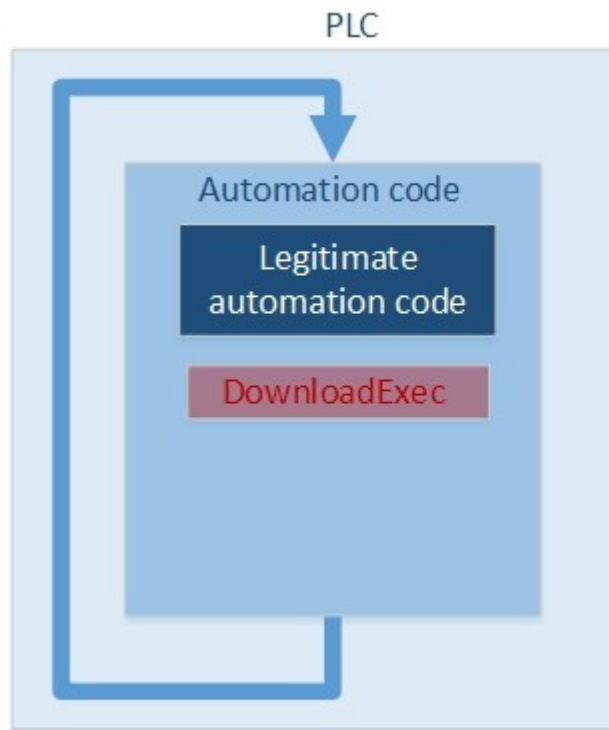
Attacker uploads
automation program



PLC backdooring

After STOP/START sequence, PLC is running “legitimate” automation program and DownloadExec backdoor

Physical automation process is not modified

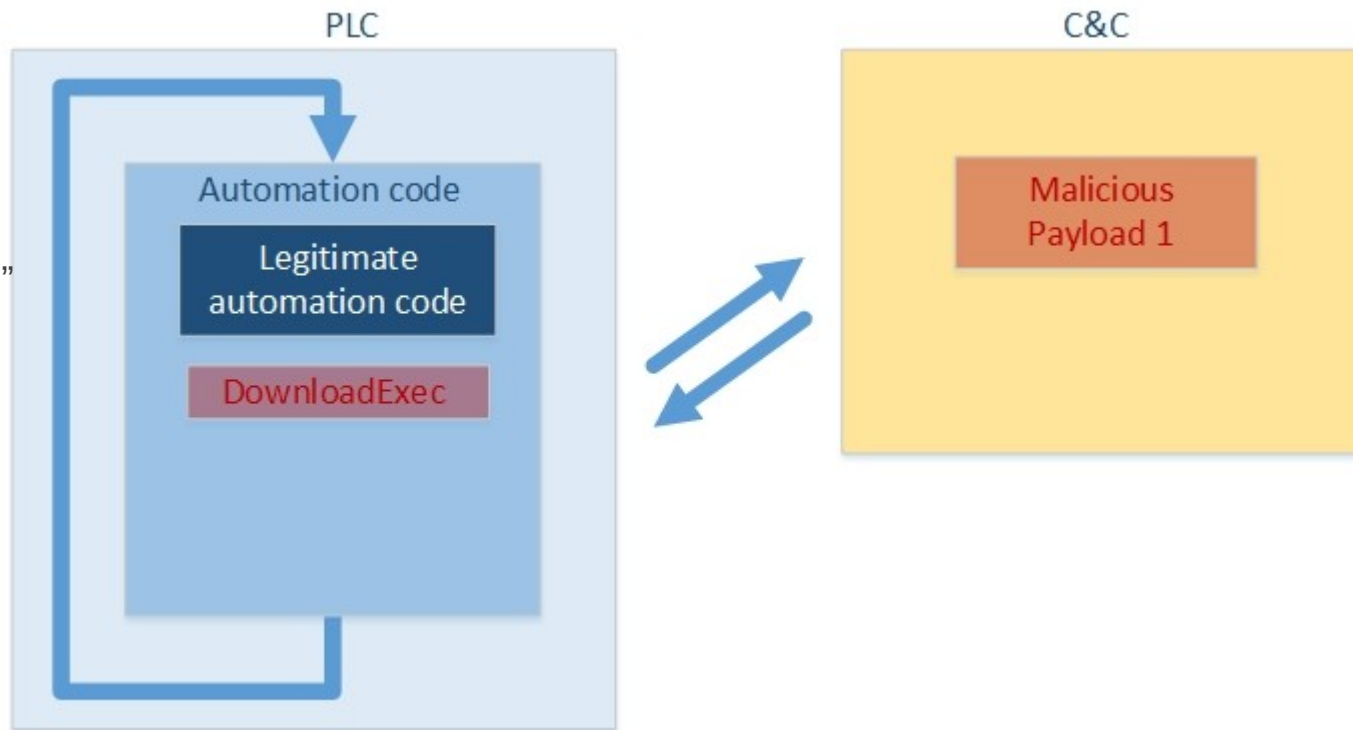


C&C



PLC backdooring

Attacker makes available
online “Malicious payload 1”
“DownloadExec” backdoor
downloads it

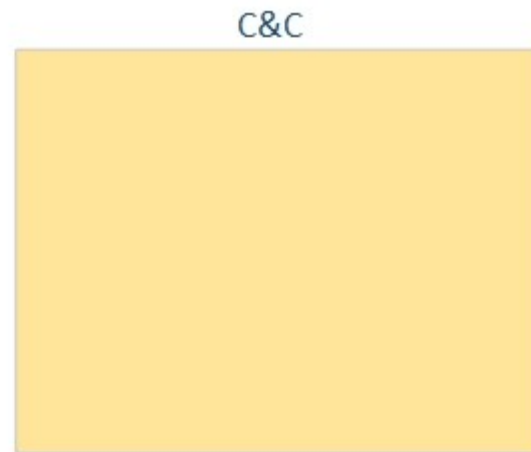
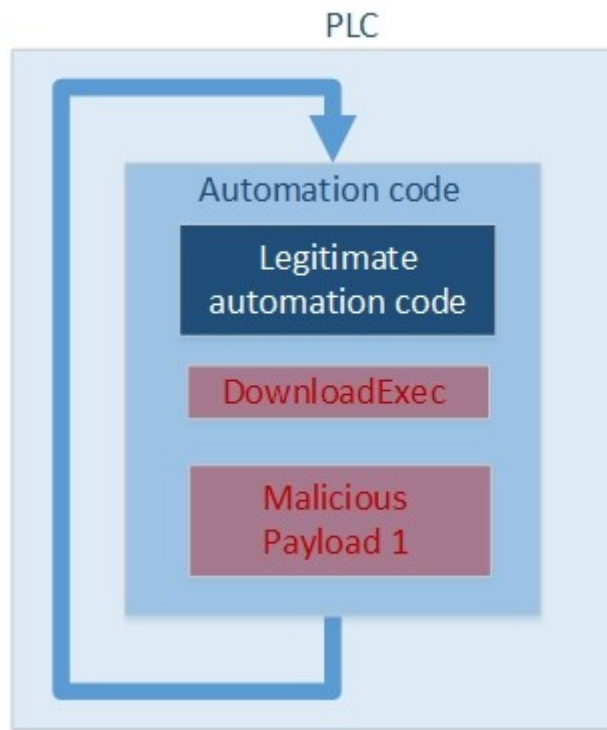


PLC backdooring

“DownloadExec” backdoor
executes “Malicious
payload 1” on-the-fly

(no more need
START/STOP sequence)

Physical automation
process can be modified

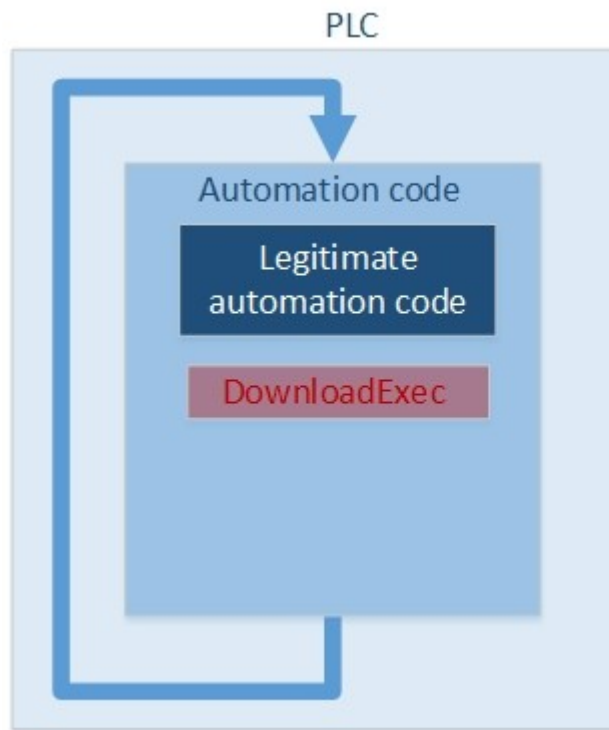


PLC backdooring

When “Malicious payload 1” execution finished, DownloadExec removes it

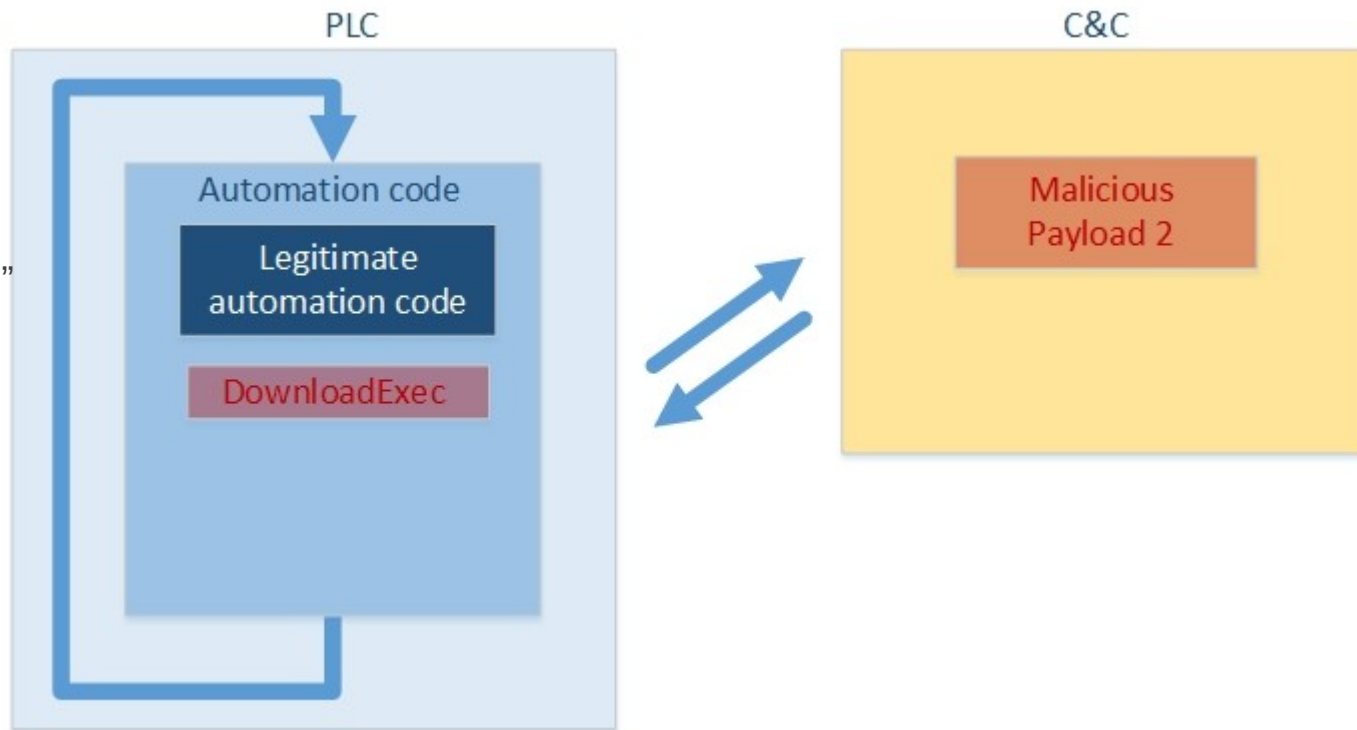
DownloadExec backdoor is waiting for another payload to execute

“Legitimate automation program” is running
□ Physical automation process is not modified



PLC backdooring

Attacker makes available
online “Malicious payload 2”
“DownloadExec” backdoor
downloads it

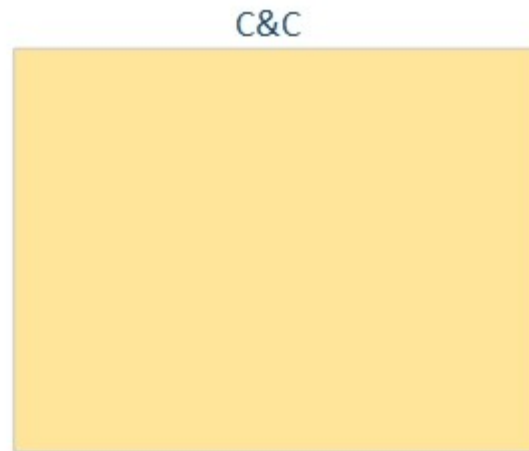
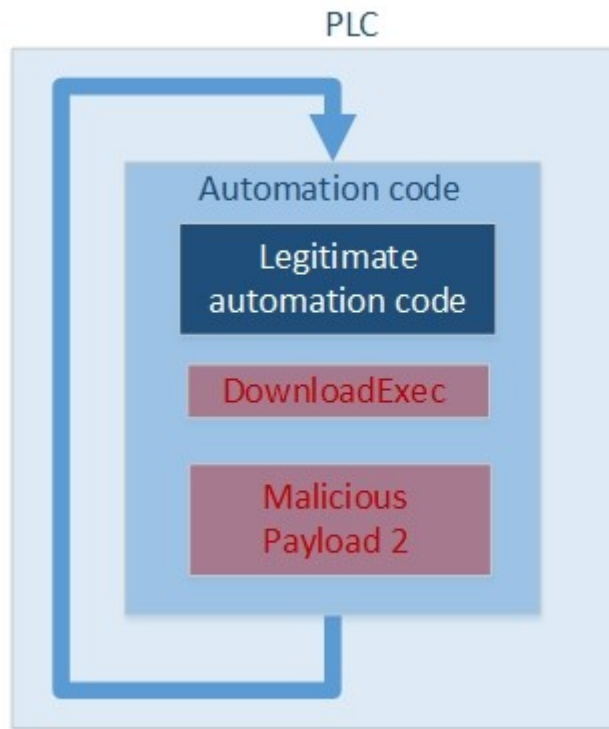


PLC backdooring

“DownloadExec” backdoor
executes “Malicious
payload 2” on-the-fly

(no more need
START/STOP sequence)

Physical automation
process can be modified

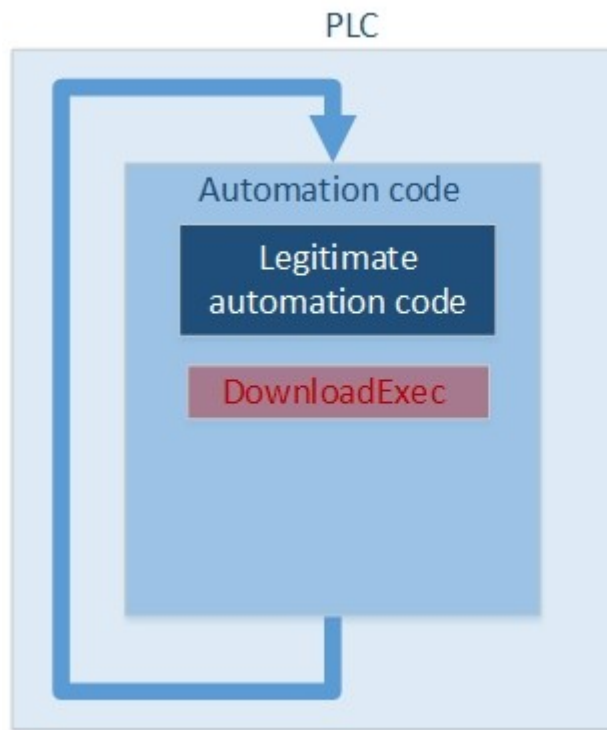


PLC backdooring

When “Malicious payload 2” execution finished, DownloadExec removes it

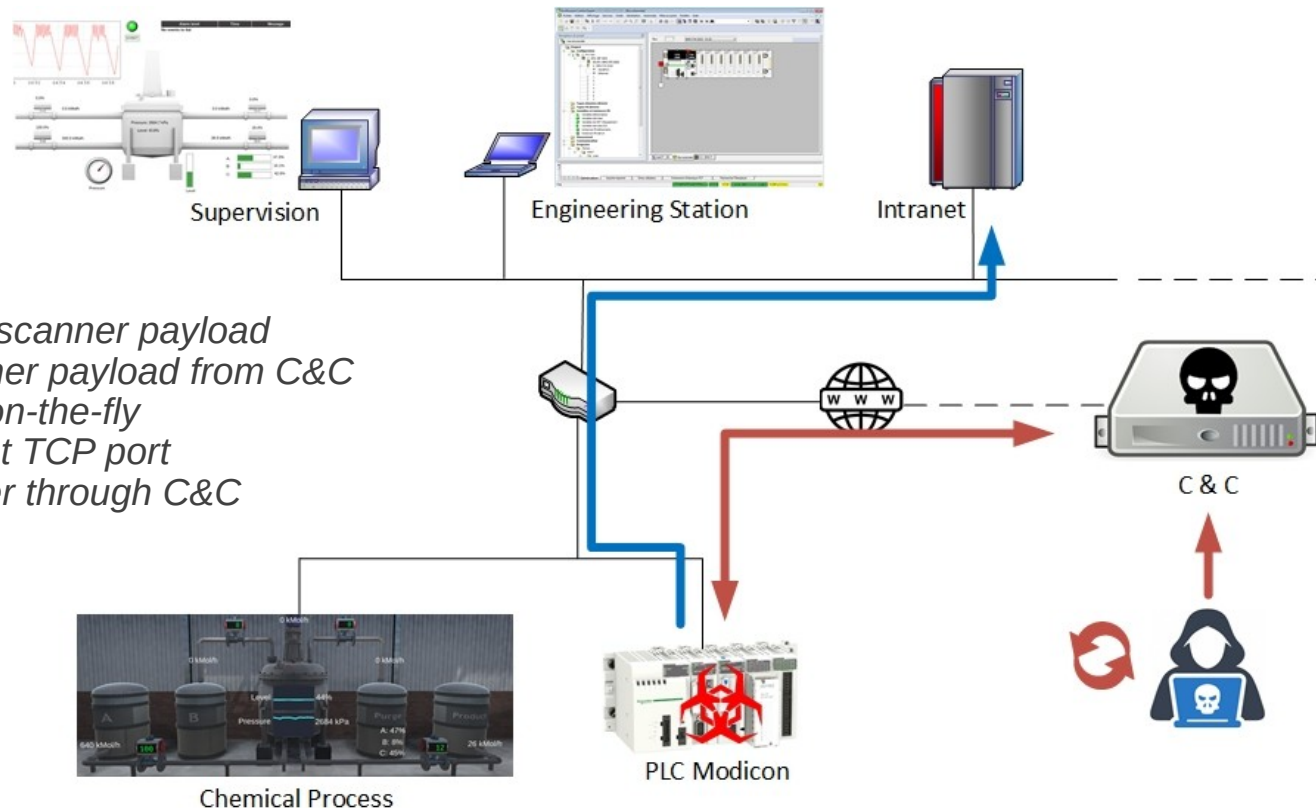
DownloadExec backdoor is waiting for another payload to execute.

“Legitimate automation program” is running
□ Physical automation process is not modified



Using PLC as a proxy

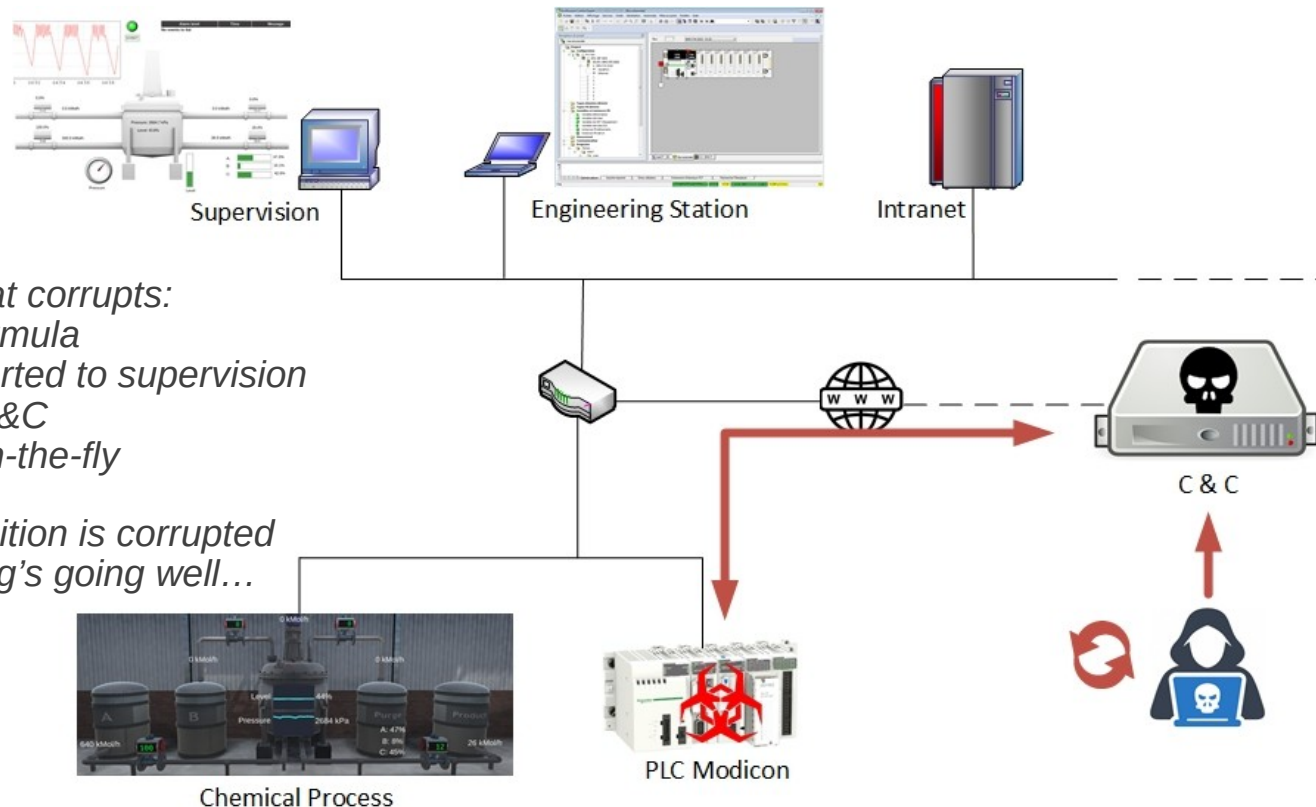
1. Attacker builds network scanner payload
2. PLC gets network scanner payload from C&C
3. PLC executes payload on-the-fly
4. PLC connects to Intranet TCP port
5. PLC reports back banner through C&C



Corrupt chemical product composition

1. Attacker builds patch that corrupts:
 - product composition formula
 - composition levels reported to supervision
2. PLC get payload from C&C
3. PLC execute payload on-the-fly

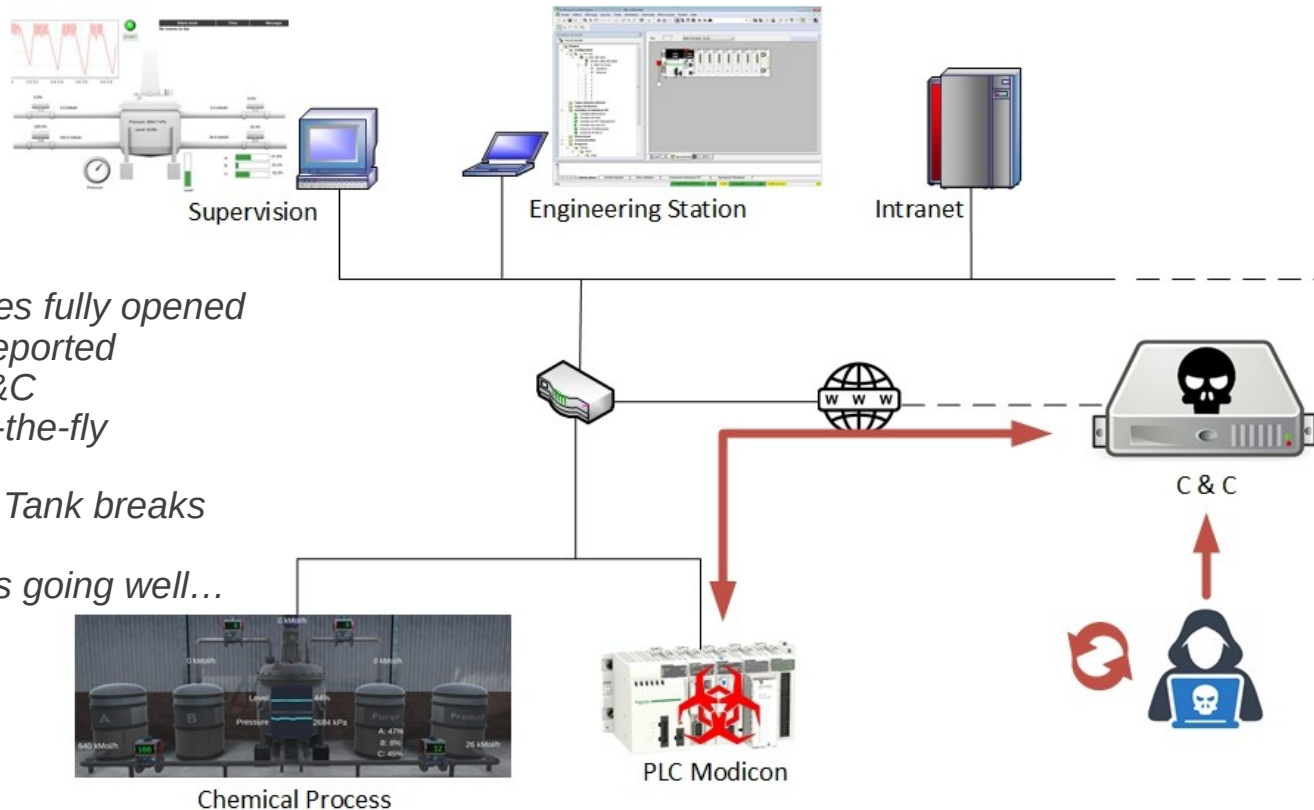
- ☐ Chemical product composition is corrupted
- ☐ For supervision, everything's going well...



Create physical damages

1. Attacker builds patch that:
 - Makes input product valves fully opened
 - Corrupts pressure level reported
2. PLC gets payload from C&C
3. PLC executes payload on-the-fly

- Pressure quickly increases. Tank breaks after few seconds
- For supervision, everything's going well...



Key takeaways

- ICS environment implies a large attack surface
 - ➔ Both Windows application and embedded systems
 - ➔ At least, following the manufacturer recommendations
- ICS manufacturers start implementing security mechanisms but...
 - ➔ Fall into same mistakes in the early 90's years
 - ➔ Architecture should be sometimes redesigned
- PLC is more than a basic electronic unit
 - ➔ Full OS underneath with all related functionalities and APIs
 - ➔ PLCs might act as a proxy forwarder, a network scanner and so on

Why not invest more on deep-dive, red or purple team services to strengthen overall security! □

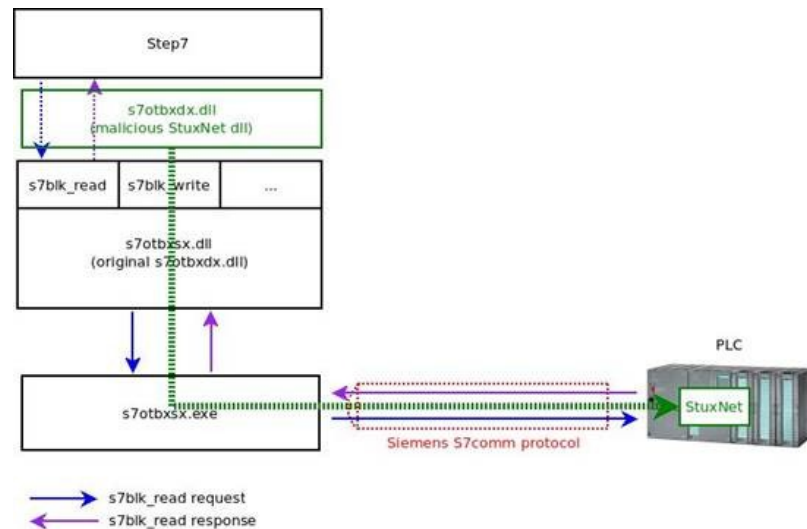
Q & A



APT in ICS

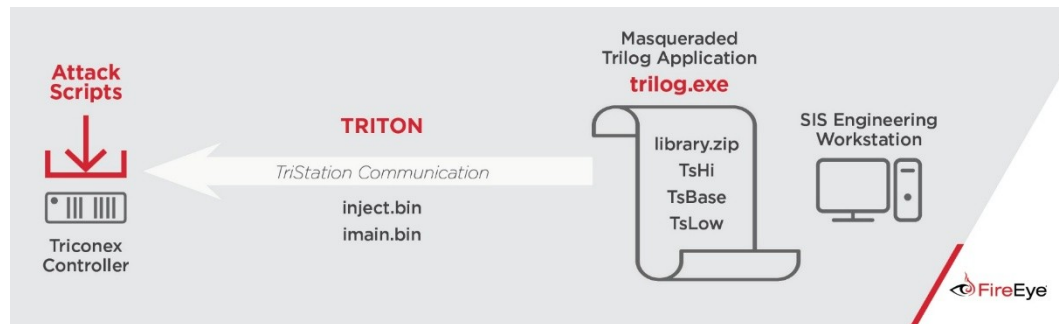
StuxNet (2010)

- Sabotage Uranium enrichment facility at Natanz/Iran
- Siemens Step7, S7-300, S7-400
- PLC payload: MC7



Triton/Trisis (2017)

- Petrochemical plant in Saudi Arabia
- Schneider Triconex (Safety controller)
- PLC payload: PowerPC



Design PLC payloads

- No debug, no print.... Emulation (Unicorn) is your friend
- Very time consuming (many weeks)
- Make a C framework to design automation program
- Add capacities to change program without START/STOP the PLC
- Transform the PLC to a proxy forwarder (C&C client)
- Hide IT/OT backdoor
 - ... without have relevant effects on the physical automation process