Kaspersky Industrial Cybersecurity Conference 2019

September 18-20, 2019, Sochi, Russia

kaspersky

The fall of CODESYS

Researching security of the framework for PLC control

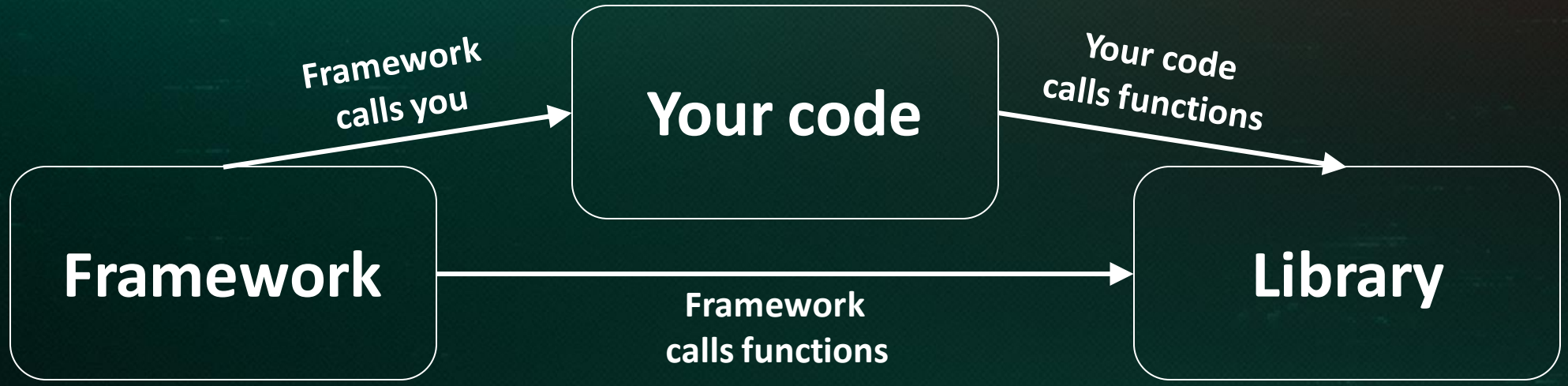
Alexander Nochvay, Security Researcher, Kaspersky

Question

What is the difference between a software library and a framework?

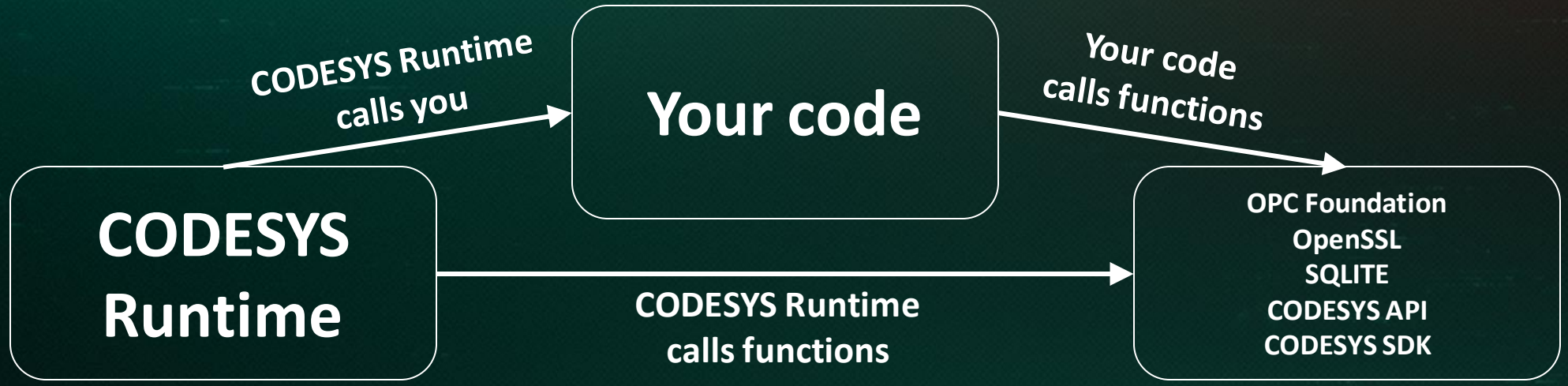
Question

What is the difference between a software library and a framework?



Question

What is inside CODESYS Runtime?



Agenda

CODESYS Runtime

Investigation protocol stack

Vulnerabilities

Conclusion

CODESYS Runtime

What is it?

CODESYS Runtime

What is it?

MCA430

emPC-X

750-8212
PFC200

>350

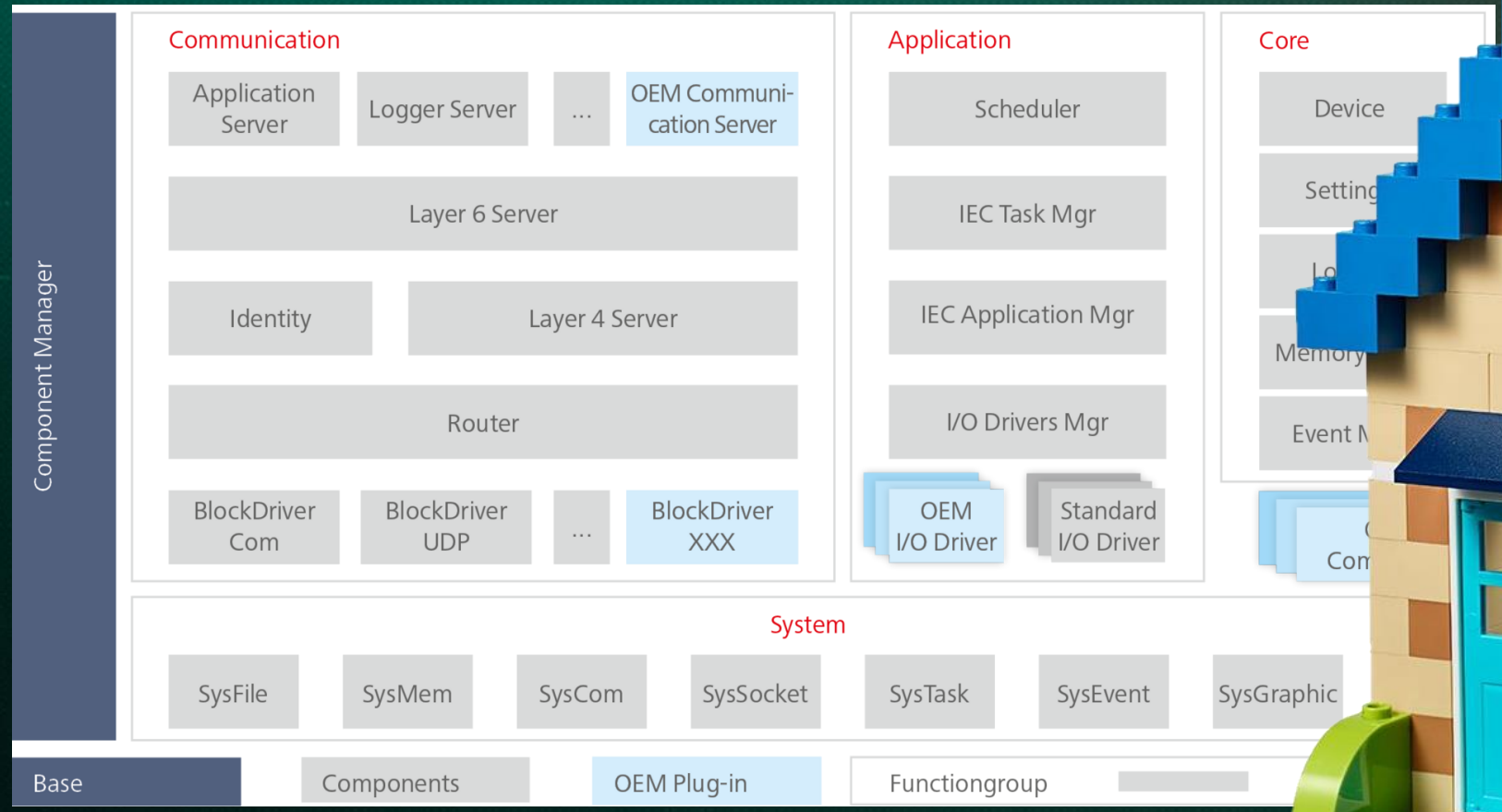
CODESYS Runtime has already been adapted

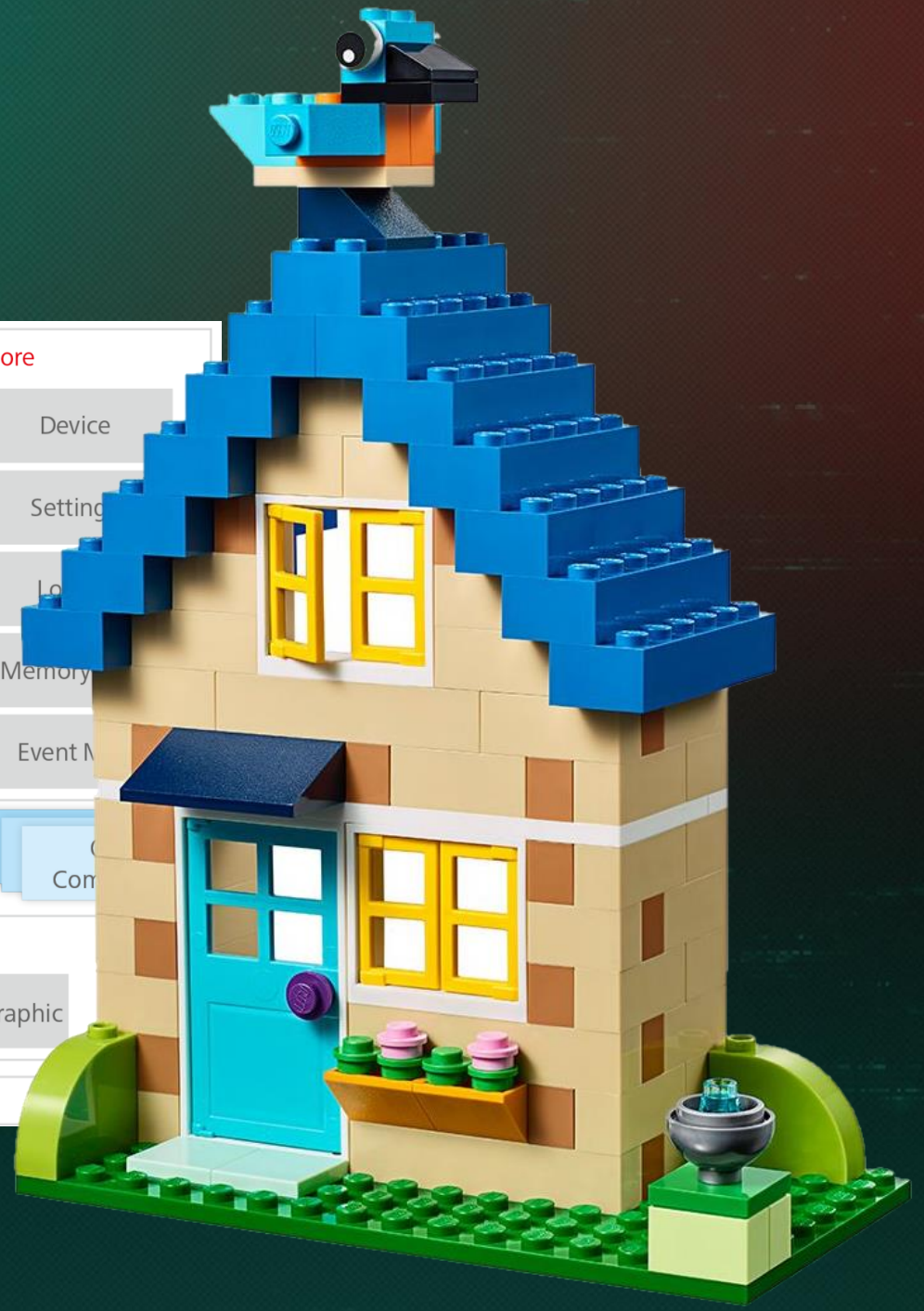
CODESYS Development System

A customizable development environment. Solution based on it include IDE:

- SoMachine by Schneider Electric
- TwinCAT by Beckhoff Automation
- IdraWorks by Bosch
- Wagilo Pro by WAGO
- CODESYS Development System by OWEN, STW Technic and prolog-plc
- And others

CODESYS Runtime Architecture





CODESYS Runtime

Components

First and main component – Component Manager

Components are dynamic libraries (like .dll, .so). Interfaces:

component->identifier

component->export_function

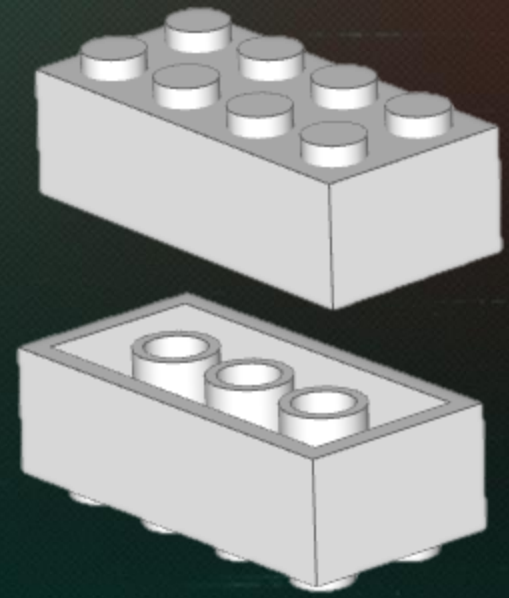
component->import_function

component->get_version

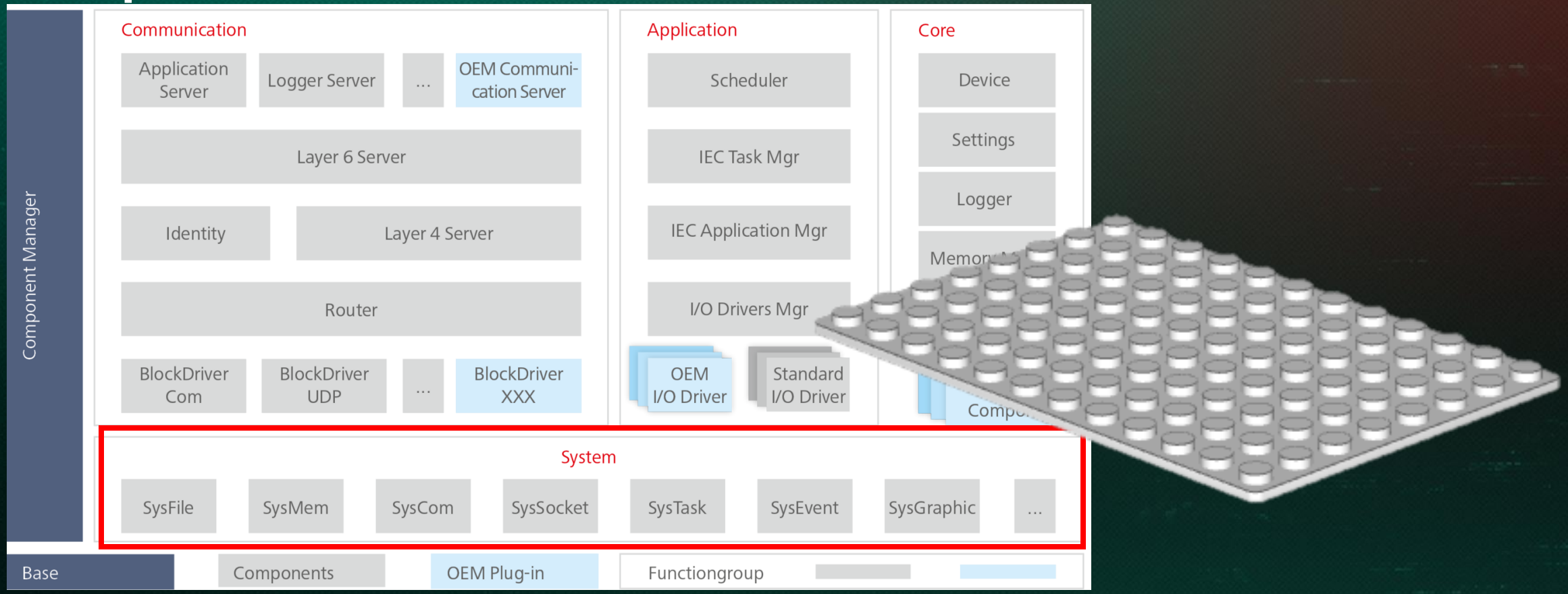
component->hook_function

component->create_instance

component->delete_isntance



CODESYS Runtime Adaptation



CODESYS Runtime

Implementation (based on CODESYS For Raspberry Pi and CODESYS For Linux)

Weak places:

- It is packed and unpacks insecure
- Insecure configuration components by default
- Run as one process
- Compiled without secure options

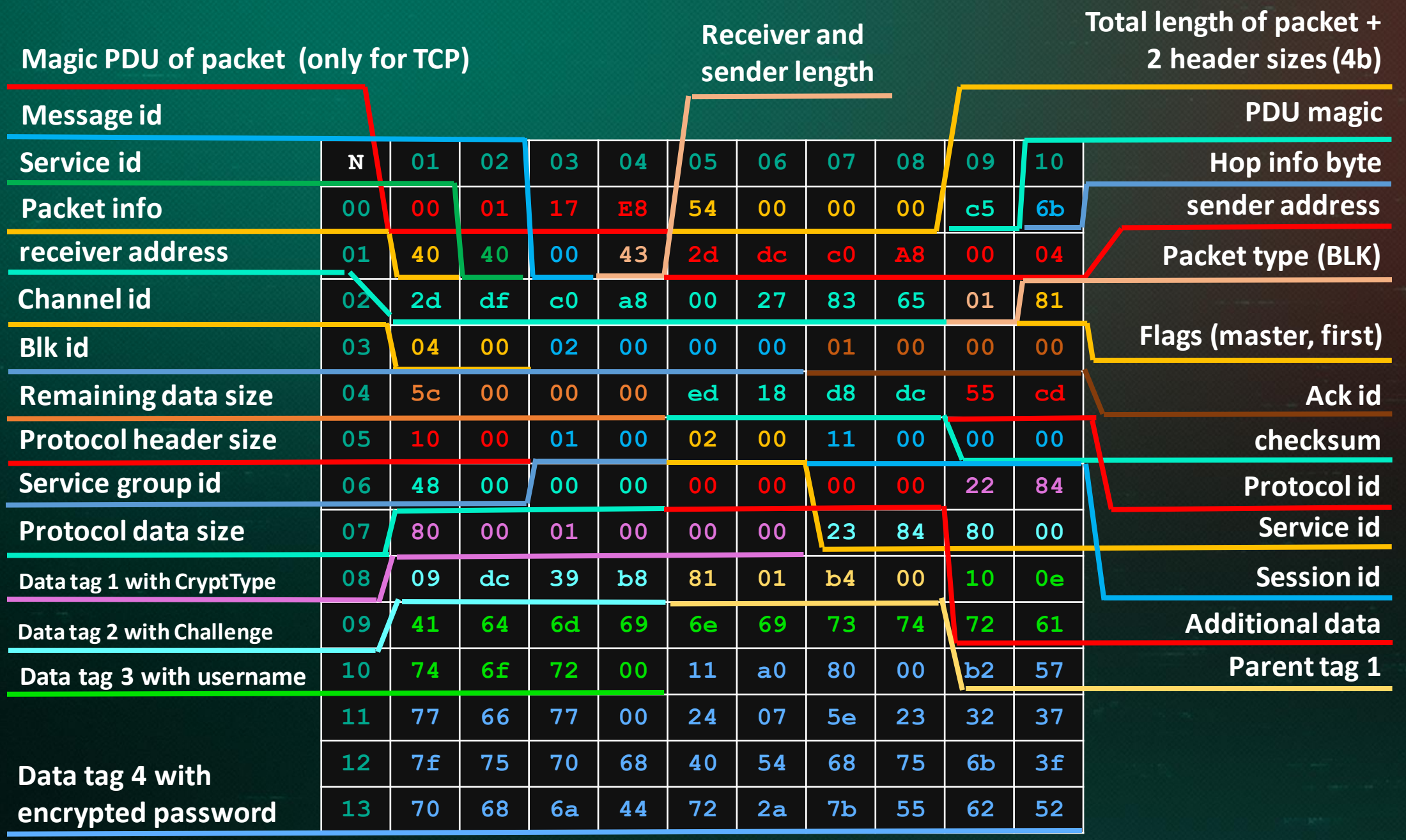
Protocol investiating

What did we find out?

CODESYS PDU (Packet data unit) Protocol
Basic description

1. It is not limited to network communication. Also for USB, CAN, serial ports
2. PDU is protocol stack consisting of four different layers: Block driver layer, Datagram layer, Channel layer and Services layer
3. PDU is based on ISO/OSI model
4. PDU sync and async type of protocol

N	01	02	03	04	05	06	07	08	09	10
00	00	01	17	E8	54	00	00	00	c5	6b
01	40	40	00	43	2d	dc	c0	A8	00	04
02	2d	df	c0	a8	00	27	83	65	01	81
03	04	00	02	00	00	00	01	00	00	00
04	5c	00	00	00	ed	18	d8	dc	55	cd
05	10	00	01	00	02	00	11	00	00	00
06	48	00	00	00	00	00	00	00	22	84
07	80	00	01	00	00	00	23	84	80	00
08	09	dc	39	b8	81	01	b4	00	10	0e
09	41	64	6d	69	6e	69	73	74	72	61
10	74	6f	72	00	11	a0	80	00	b2	57
11	77	66	77	00	24	07	5e	23	32	37
12	7f	75	70	68	40	54	68	75	6b	3f
13	70	68	6a	44	72	2a	7b	55	62	52

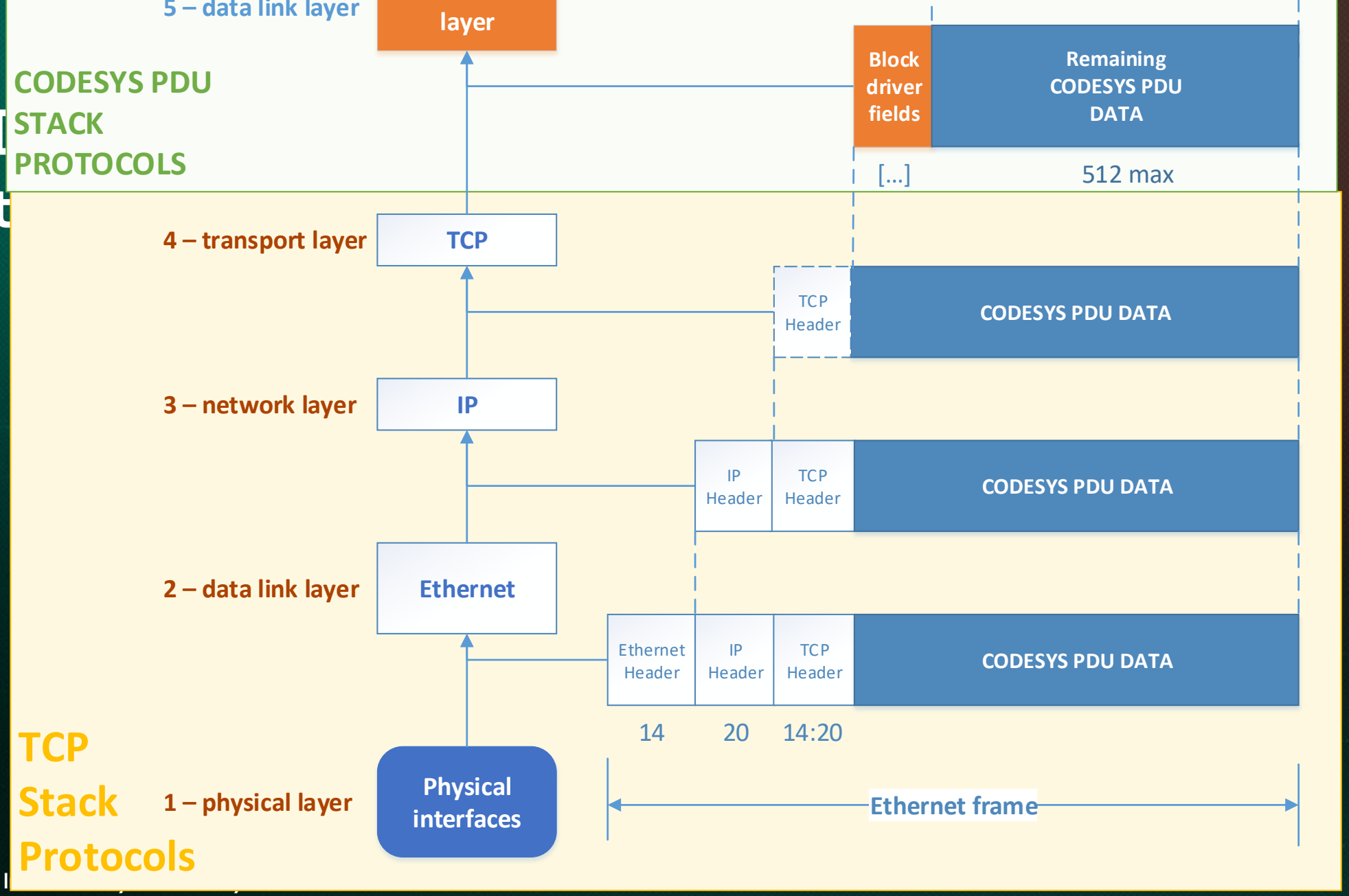

Magic PDU of packet (only for TCP)
Receiver and sender length
Total length of packet + 2 header sizes (4b)
Message id
Service id
Packet info
receiver address
Channel id
Blk id
Remaining data size
Protocol header size
Service group id
Protocol data size
Data tag 1 with CryptType
Data tag 2 with Challenge
Data tag 3 with username
Data tag 4 with encrypted password
PDU magic
Hop info byte
sender address
Packet type (BLK)
Flags (master, first)
Ack id
checksum
Protocol id
Service id
Session id
Additional data
Parent tag 1
N 01 02 03 04 05 06 07 08 09 10
00 00 01 17 E8 54 00 00 00 c5 6b
01 40 40 00 43 2d dc c0 A8 00 04
02 2d df c0 a8 00 27 83 65 01 81
03 04 00 02 00 00 00 01 00 00 00
04 5c 00 00 00 ed 18 d8 dc 55 cd
05 10 00 01 00 02 00 11 00 00 00
06 48 00 00 00 00 00 00 00 22 84
07 80 00 01 00 00 00 23 84 80 00
08 09 dc 39 b8 81 01 b4 00 10 0e
09 41 64 6d 69 6e 69 73 74 72 61
10 74 6f 72 00 11 a0 80 00 b2 57
11 77 66 77 00 24 07 5e 23 32 37
12 7f 75 70 68 40 54 68 75 6b 3f
13 70 68 6a 44 72 2a 7b 55 62 52

CODESYS PDU (Packet data unit) Protocol

Fact #1. One ISO/OSI is good. Two is ~~better~~

Kaspersky Industrial Cybersecurity Conference 2019

COD
Fact

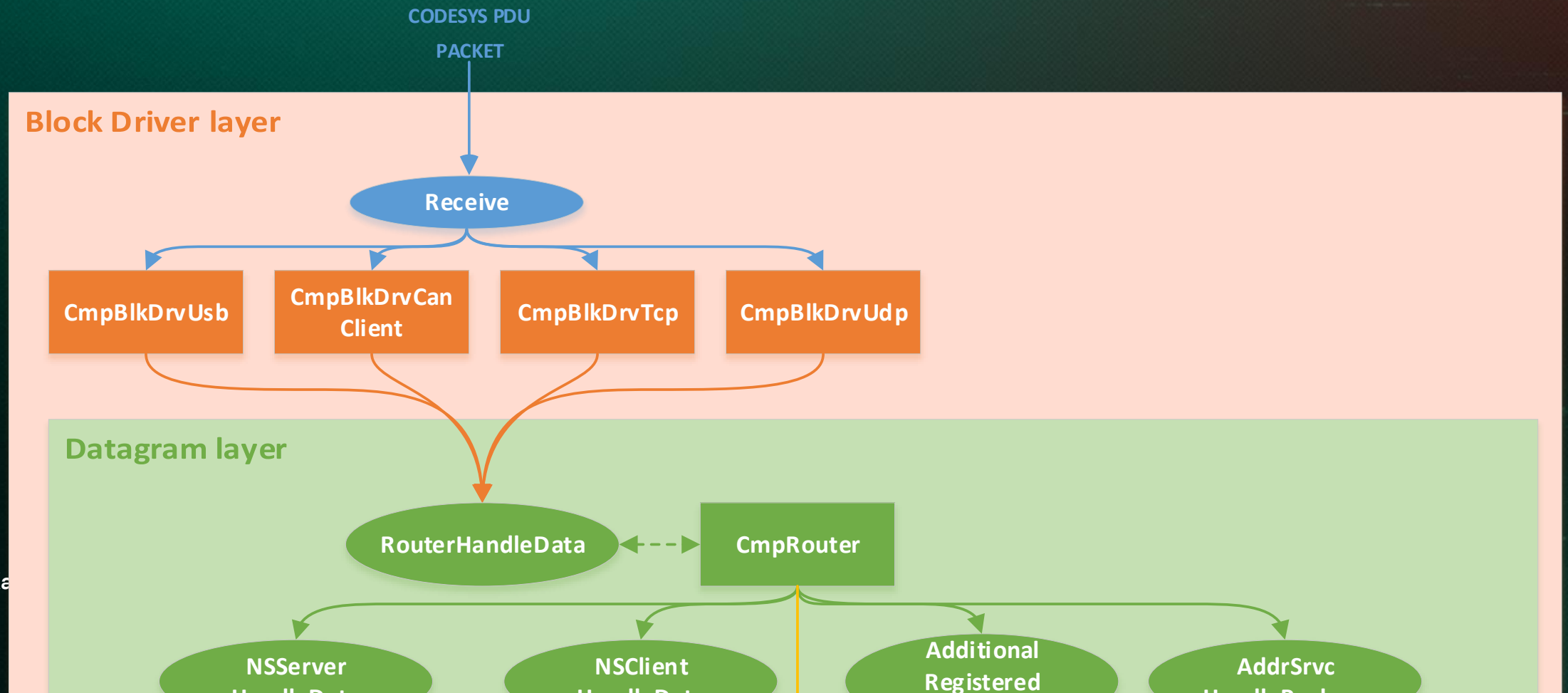

5 – data link layer
layer
CODESYS PDU
STACK
PROTOCOLS
Block driver fields
Remaining CODESYS PDU DATA
[...]
512 max
4 – transport layer
TCP
TCP Header
CODESYS PDU DATA
3 – network layer
IP
IP Header
TCP Header
CODESYS PDU DATA
2 – data link layer
Ethernet
Ethernet Header
IP Header
TCP Header
CODESYS PDU DATA
14
20
14:20
TCP
Stack
Protocols
1 – physical layer
Physical interfaces
Ethernet frame

CODESYS PDU (Packet data unit) Protocol

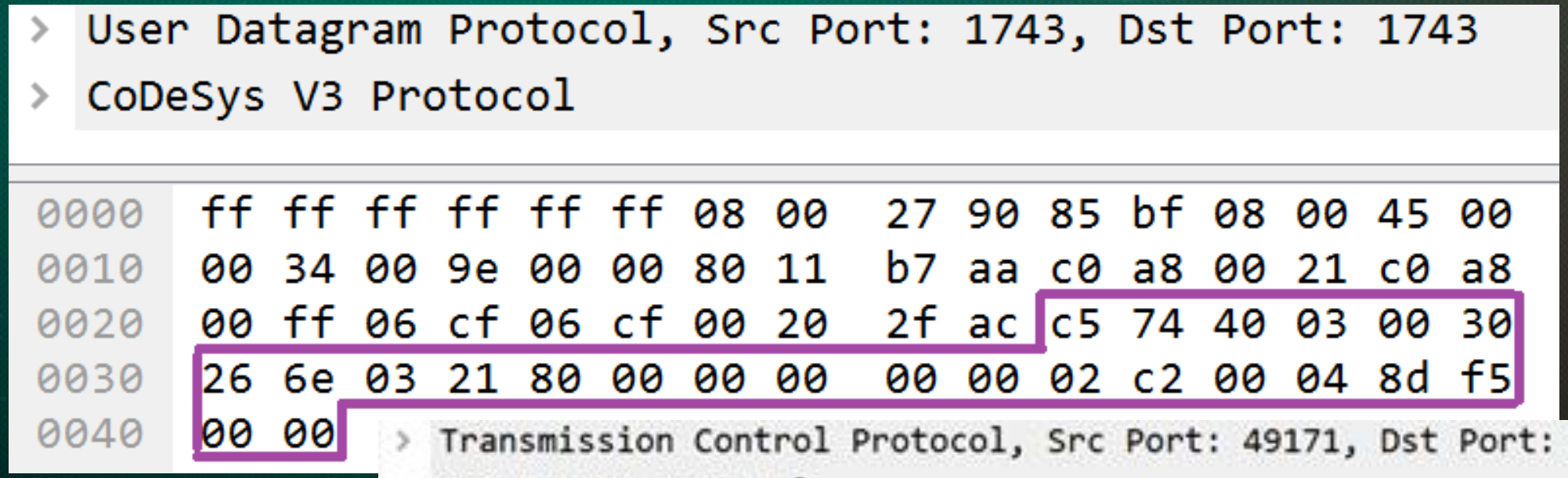
Fact #2. More than 10 components process the network packet

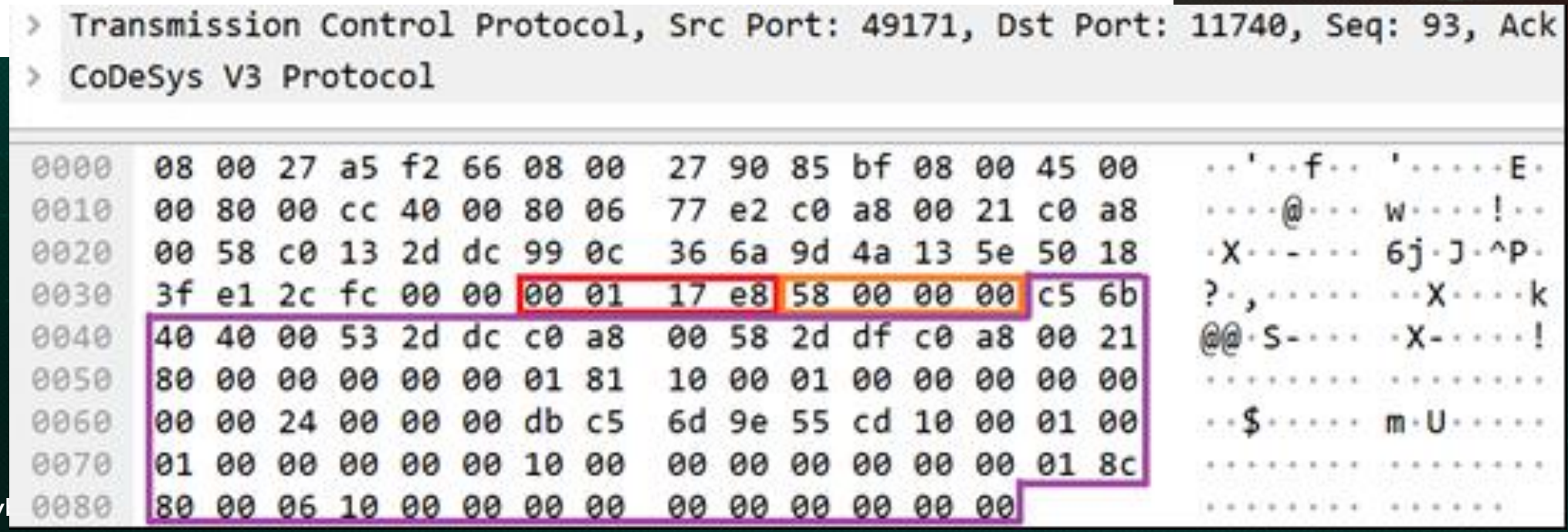
CODESYS PDU (Packet data unit) Protocol

Fact #2. More than 10 components process the network packet



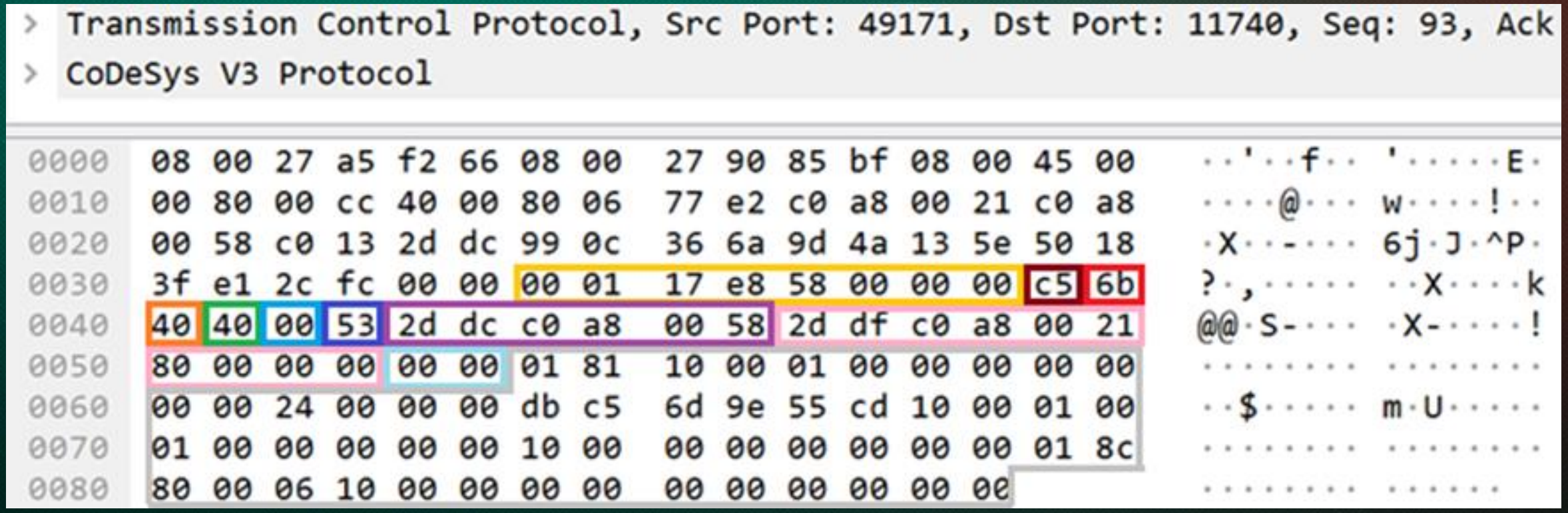
CODESYS PDU (Packet data unit) Protocol
Fact #3. Block components add additional fields


User Datagram Protocol, Src Port: 1743, Dst Port: 1743
CoDeSys V3 Protocol
0000 ff ff ff ff ff ff 08 00 27 90 85 bf 08 00 45 00
0010 00 34 00 9e 00 00 80 11 b7 aa c0 a8 00 21 c0 a8
0020 00 ff 06 cf 06 cf 00 20 2f ac c5 74 40 03 00 30
0030 26 6e 03 21 80 00 00 00 00 00 02 c2 00 04 8d f5
0040 00 00


Transmission Control Protocol, Src Port: 49171, Dst Port: 11740, Seq: 93, Ack
CoDeSys V3 Protocol
0000 08 00 27 a5 f2 66 08 00 27 90 85 bf 08 00 45 00
0010 00 80 00 cc 40 00 80 06 77 e2 c0 a8 00 21 c0 a8
0020 00 58 c0 13 2d dc 99 0c 36 6a 9d 4a 13 5e 50 18
0030 3f e1 2c fc 00 00 00 01 17 e8 58 00 00 00 c5 6b
0040 40 40 00 53 2d dc c0 a8 00 58 2d df c0 a8 00 21
0050 80 00 00 00 00 00 01 81 10 00 01 00 00 00 00 00
0060 00 00 24 00 00 00 db c5 6d 9e 55 cd 10 00 01 00
0070 01 00 00 00 00 00 10 00 00 00 00 00 00 00 01 8c
0080 80 00 06 10 00 00 00 00 00 00 00 00 00 00

CODESYS PDU (Packet data unit) Protocol

Fact #4. CODESYS PDU packet contains addresses of sender and receiver



Color								
fields	service_id	message_id	lengths		sender		Receiver	
			receiver_ length	sender_ length	port	address	port	Address
value	0x40	0x00	0x5	0x3	11740 (2ddc)	192.168.0.88 (c0a80058)	11743 (2ddf)	192.168.0.33 (c0a80021) 800000

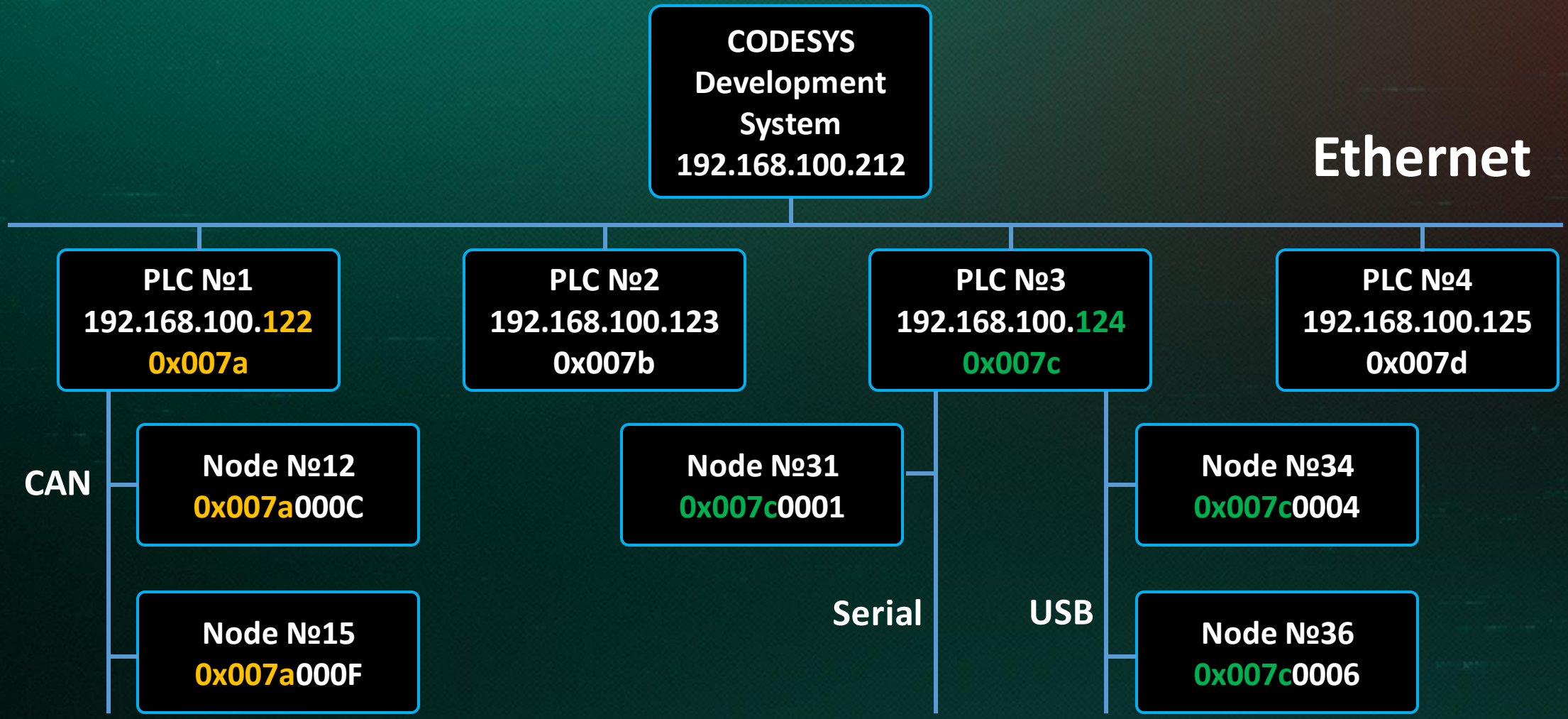
```
> User Datagram Protocol, Src Port: 1743, Dst Port: 1740
> CoDeSys V3 Protocol
```

```
0000  08 00 27 a5 f2 66 08 00  27 90 85 bf 08 00 45 00   ··'··f·· '·····E·
0010  00 64 15 df 40 00 80 11  62 e0 c0 a8 00 21 c0 a8   ·d··@··· b····!··
0020  00 58 06 cf 06 cc 00 50  53 60 c5 6b 40 40 00 31   ·X·····P S`·k@@·1
0030  00 58 03 21 80 00 00 00  00 00 01 81 24 00 01 00   ·X·!···· ····$···
0040  00 00 00 00 00 00 24 00  00 00 13 fd 2b 3a 55 cd   ·····$·· ····+:U·
0050  10 00 01 00 01 00 00 00  00 00 10 00 00 00 00 00   ···· ···· ····
0060  00 00 01 8c 80 00 06 10  00 00 05 00 00 00 00 0d   ···· ···· ····
0070  05 03                                             ··
```

Color								
fields	lengths		Sender		receiver		padding (optional)	Remaining data
	receiver_ length	sender_ length	port_ index	relative address	port_ index	relative_ address		
value	0x1	0x3	0	88 (0x58)	3	33 (0x21) 800000	0x0000	[...]

CODESYS PDU (Packet data unit) Protocol

Fact #4. CODESYS PDU packet contains addresses of sender and receiver



CODESYS PDU (Packet data unit) Protocol

Fact #5. Following components are identified as services

CmpApp – 0x2

CmpAlarmManager – 0x18

CmpAppBP – 0x12

CmpAppForce – 0x13

CmpCodeMeter – 0x1d

CmpCoreDump – 0x1f

CmpDevice – 0x1

CmpFileTransfer – 0x8

CmpIecVarAccess – 0x9

CmpIoMgr – 0xb

CmpLog – 0x5

CmpMonitor – 0x1b

CmpOpenSSL – 0x22

CmpSettings – 0x6

CmpTraceMgr – 0xf

CmpTraceMgr – 0xf

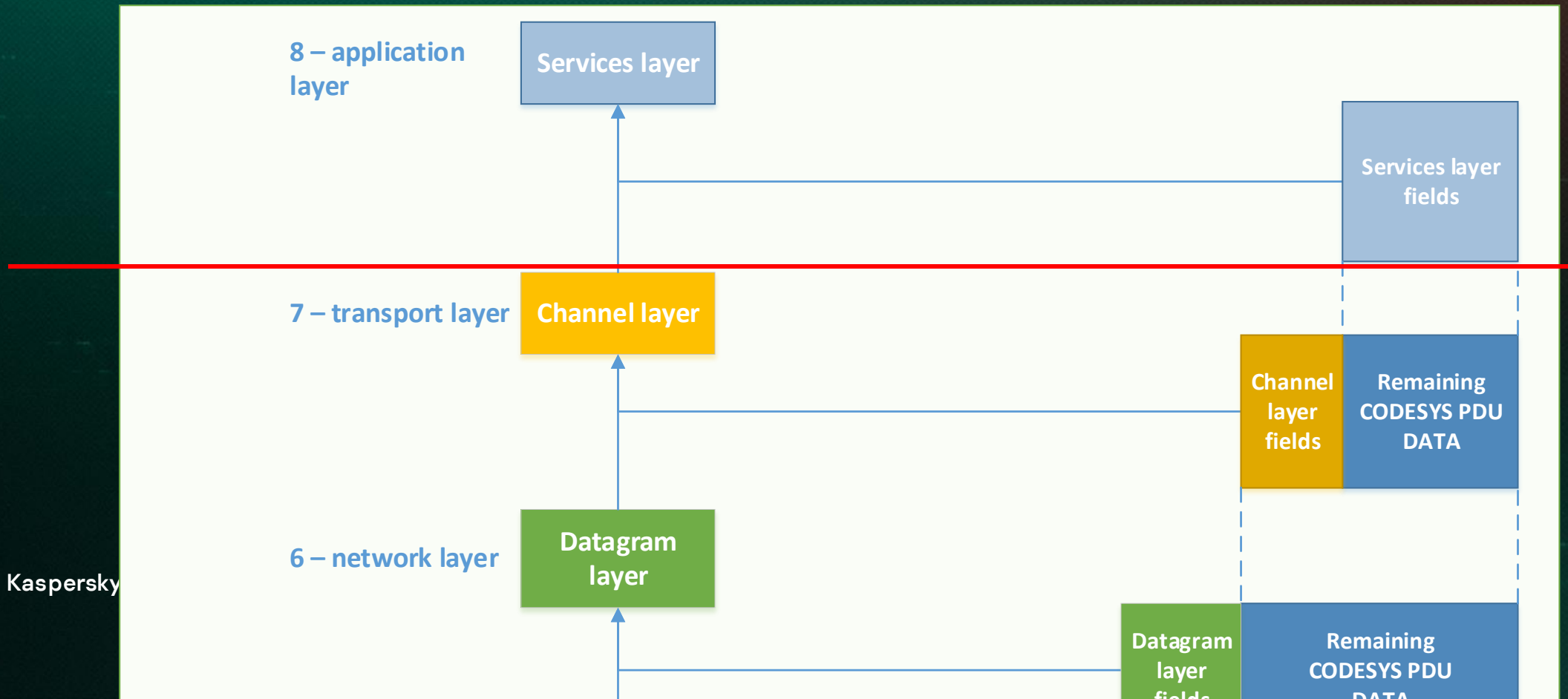
CmpUserMgr – 0xc

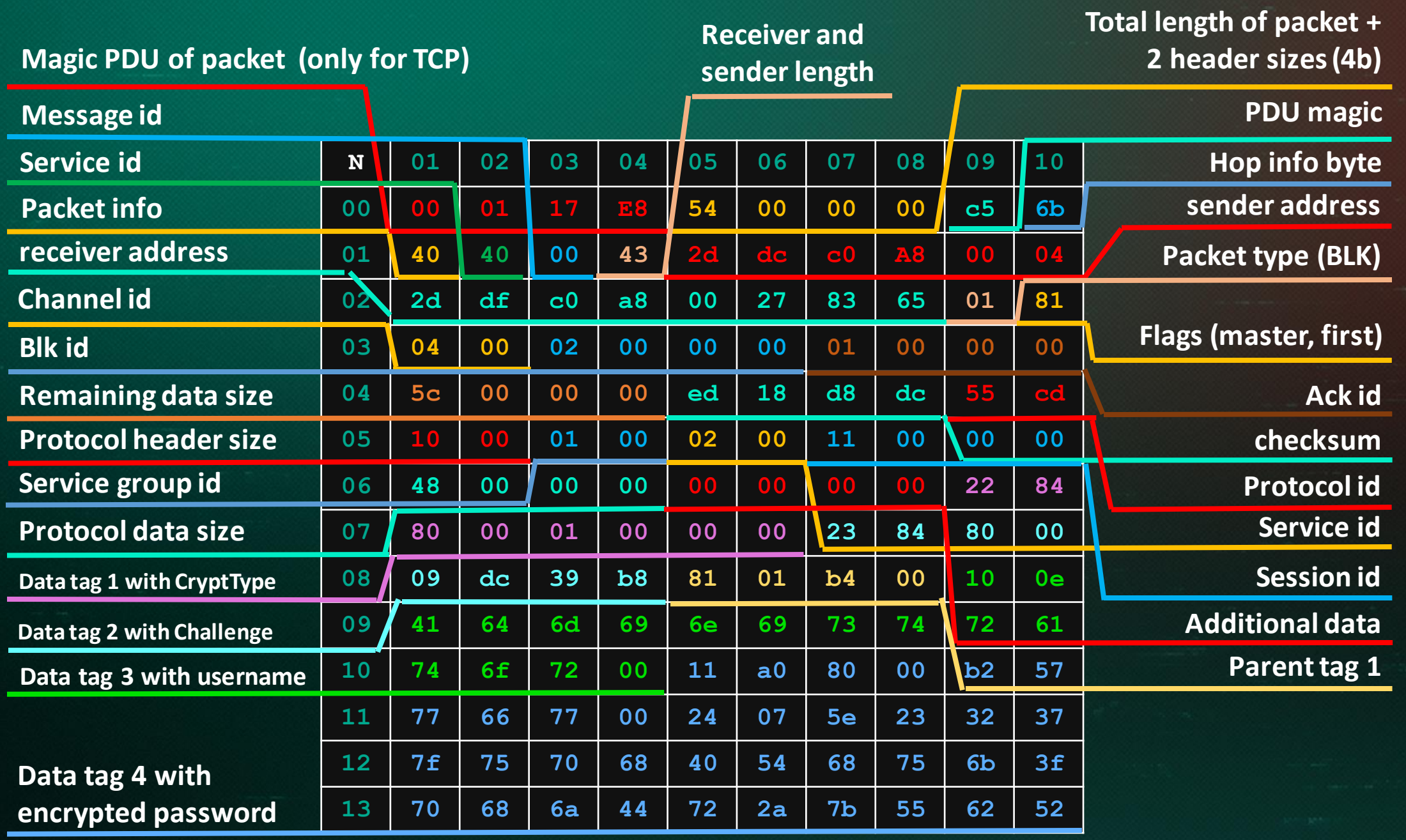
CmpVisuServer – 0x4

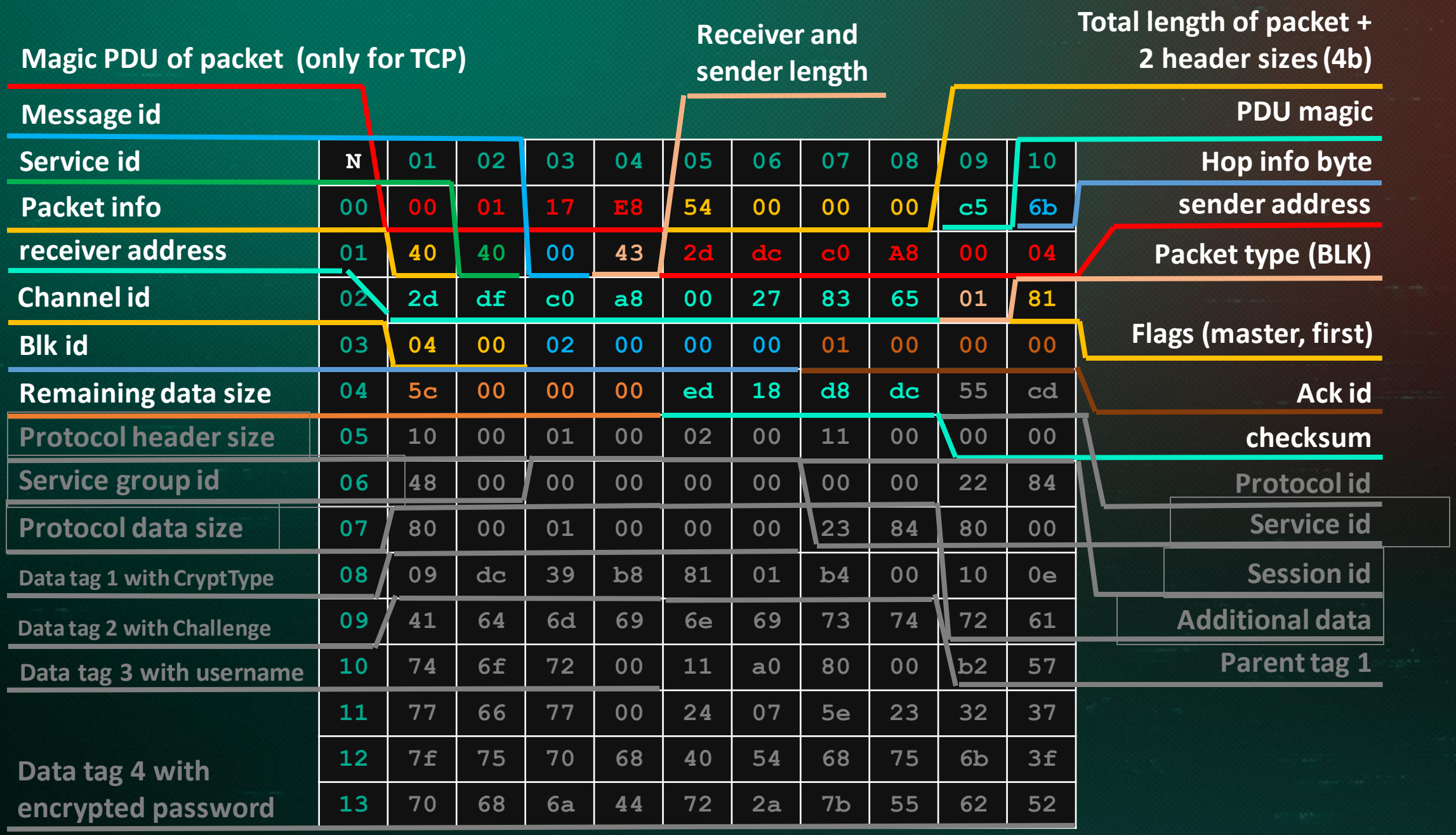
PlcShell – 0x11

SysEthernet – 0x7

CODESYS PDU (Packet data unit) Protocol
Fact #6. Enabled encryption between PLC and IDE encrypts only service layer




Magic PDU of packet (only for TCP)
Receiver and sender length
Total length of packet + 2 header sizes (4b)
Message id
Service id
Packet info
receiver address
Channel id
Blk id
Remaining data size
Protocol header size
Service group id
Protocol data size
Data tag 1 with CryptType
Data tag 2 with Challenge
Data tag 3 with username
Data tag 4 with encrypted password
PDU magic
Hop info byte
sender address
Packet type (BLK)
Flags (master, first)
Ack id
checksum
Protocol id
Service id
Session id
Additional data
Parent tag 1
N 01 02 03 04 05 06 07 08 09 10
00 00 01 17 E8 54 00 00 00 c5 6b
01 40 40 00 43 2d dc c0 A8 00 04
02 2d df c0 a8 00 27 83 65 01 81
03 04 00 02 00 00 00 01 00 00 00
04 5c 00 00 00 ed 18 d8 dc 55 cd
05 10 00 01 00 02 00 11 00 00 00
06 48 00 00 00 00 00 00 00 22 84
07 80 00 01 00 00 00 23 84 80 00
08 09 dc 39 b8 81 01 b4 00 10 0e
09 41 64 6d 69 6e 69 73 74 72 61
10 74 6f 72 00 11 a0 80 00 b2 57
11 77 66 77 00 24 07 5e 23 32 37
12 7f 75 70 68 40 54 68 75 6b 3f
13 70 68 6a 44 72 2a 7b 55 62 52

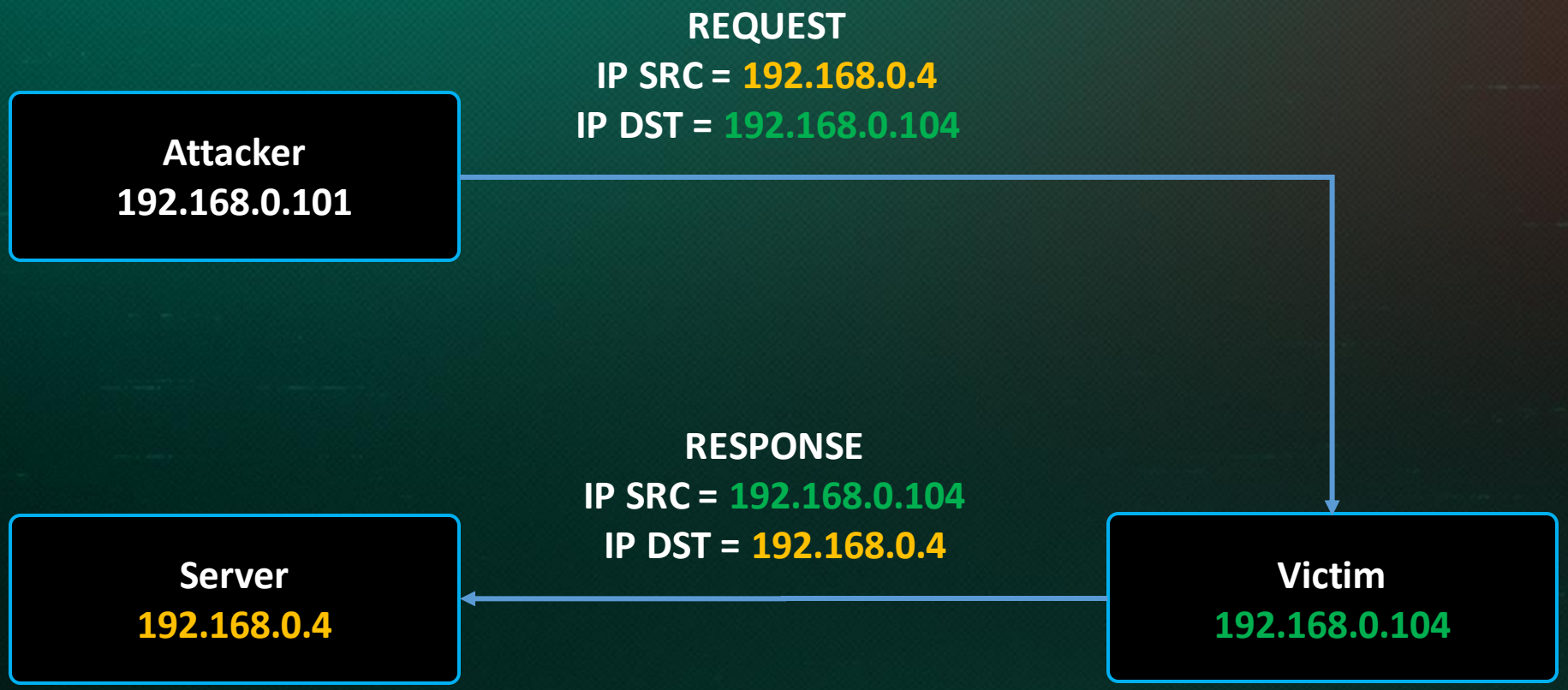

Magic PDU of packet (only for TCP)
Receiver and sender length
Total length of packet + 2 header sizes (4b)
Message id
PDU magic
Service id
Hop info byte
Packet info
sender address
receiver address
Packet type (BLK)
Channel id
Flags (master, first)
Blk id
Ack id
Remaining data size
checksum
Protocol header size
Protocol id
Service group id
Service id
Protocol data size
Session id
Data tag 1 with CryptType
Additional data
Data tag 2 with Challenge
Parent tag 1
Data tag 3 with username
Data tag 4 with encrypted password
N 01 02 03 04 05 06 07 08 09 10
00 00 01 17 E8 54 00 00 00 c5 6b
01 40 40 00 43 2d dc c0 A8 00 04
02 2d df c0 a8 00 27 83 65 01 81
03 04 00 02 00 00 00 01 00 00 00
04 5c 00 00 00 ed 18 d8 dc 55 cd
05 10 00 01 00 02 00 11 00 00 00
06 48 00 00 00 00 00 00 00 22 84
07 80 00 01 00 00 00 23 84 80 00
08 09 dc 39 b8 81 01 b4 00 10 0e
09 41 64 6d 69 6e 69 73 74 72 61
10 74 6f 72 00 11 a0 80 00 b2 57
11 77 66 77 00 24 07 5e 23 32 37
12 7f 75 70 68 40 54 68 75 6b 3f
13 70 68 6a 44 72 2a 7b 55 62 52

Vulnerabilities

Plan: looking for inherited the shortcomings of model ISO/OSI

Vulnerability #1. Address spoofing

Classic IP-Spoofing



Vulnerability #1. Address spoofing

CODESYS address spoofing. Type #1 - Classic

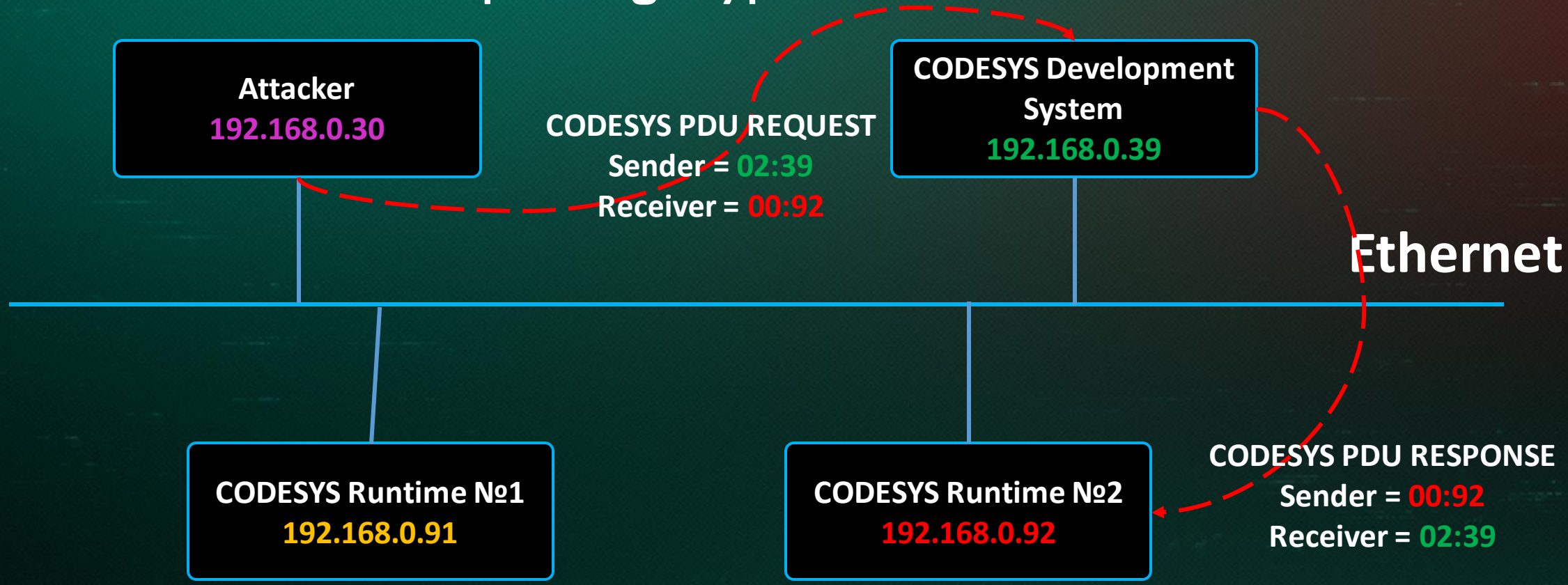
```
> Frame 8: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interfa
> Ethernet II, Src: PcsCompu_90:85:bf (08:00:27:90:85:bf), Dst: Raspberr_92:3a:
> Internet Protocol Version 4, Src: 192.168.0.39, Dst: 192.168.0.92
> User Datagram Protocol, Src Port: 1742, Dst Port: 1740
> CoDeSys V3 Protocol
```

```
0000  b8 27 eb 92 3a ff 08 00  27 90 85 bf 08 00 45 00   ·'··:··· '·····E·
0010  00 3c 02 03 00 00 80 11  b6 da c0 a8 00 27 c0 a8   ·<·····  ·····'··
0020  00 5c 06 ce 06 cc 00 28  a1 f8 e5 72 40 40 00 11   ·\·····(  ···r@@··
0030  00 5c 02 27 00 00 c3 00  (                        ·\·'····  (
0040  37 39 00 40 1f 00 04 00  (                        79·@····  (
```

Color				
fields	Sender		**Receiver**	
	Port index	Address	Port index	Address
value	0 (1740)	0x5c (192.168.0.92)	2 (1742)	0x27 (192.168.0.39)

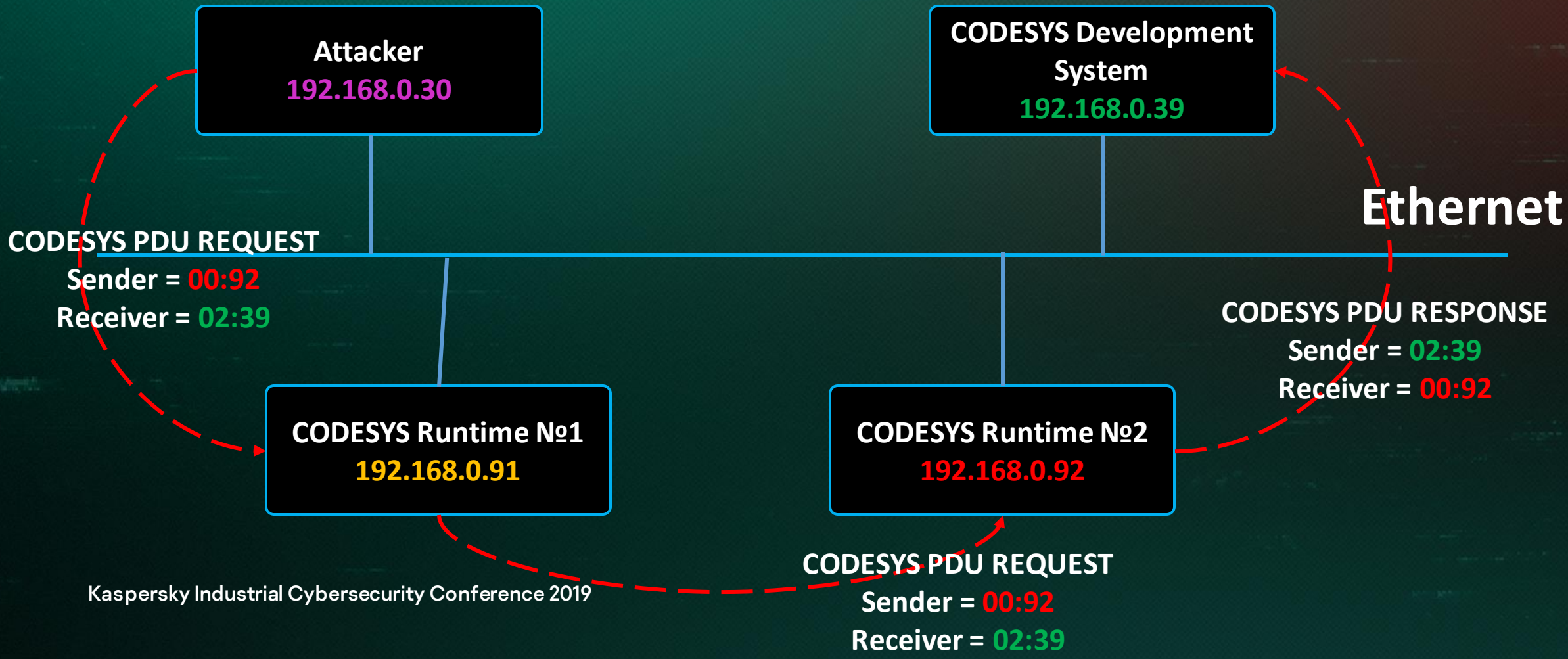
Vulnerability #1. Address spoofing

CODESYS address spoofing. Type #1 - Classic



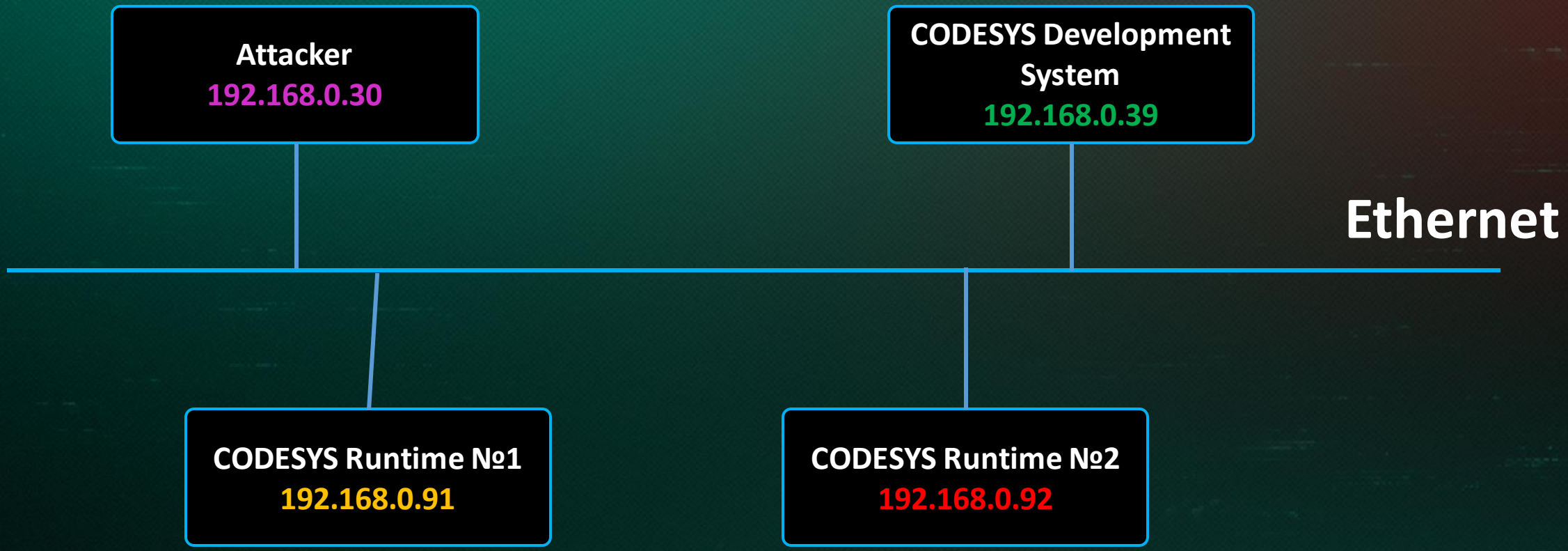
Vulnerability #1. Address spoofing

CODESYS address spoofing. Type #2 – Modified classic



Vulnerability #1. Address spoofing

CODESYS address spoofing. Type #3 – With concealed receipt of a response to a request



```
> Frame 1444: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
> Ethernet II, Src: D-LinkIn_ad:fb:c0 (9c:d6:43:ad:fb:c0), Dst: Raspberr_92:3a:ff (b8:27:eb:92:3a:ff)
> Internet Protocol Version 4, Src: 192.168.0.30, Dst: 192.168.0.92
> User Datagram Protocol, Src Port: 1740, Dst Port: 1740
> CoDeSys V3 Protocol
```

```
0000  b8 27 eb 92 3a ff 9c d6  43 ad fb c0 08 00 45 00   ·'··:··· C·····E·
0010  00 3c f6 09 00 00 80 11  00 00 c0 a8 00 1e c0 a8   ·<·····  ········
0020  00 5c 06 cc 06 cc 00 28  82 04 c5 73 40 40 00 11   ·\·····(  ···s@@··
0030  00 5c 00 ff 00 00 c3 00  01 01 bf 19 68 af 00 00   ·\·····  ····h···
0040  00 00 00 40 1f 00 04 00  00 00
```

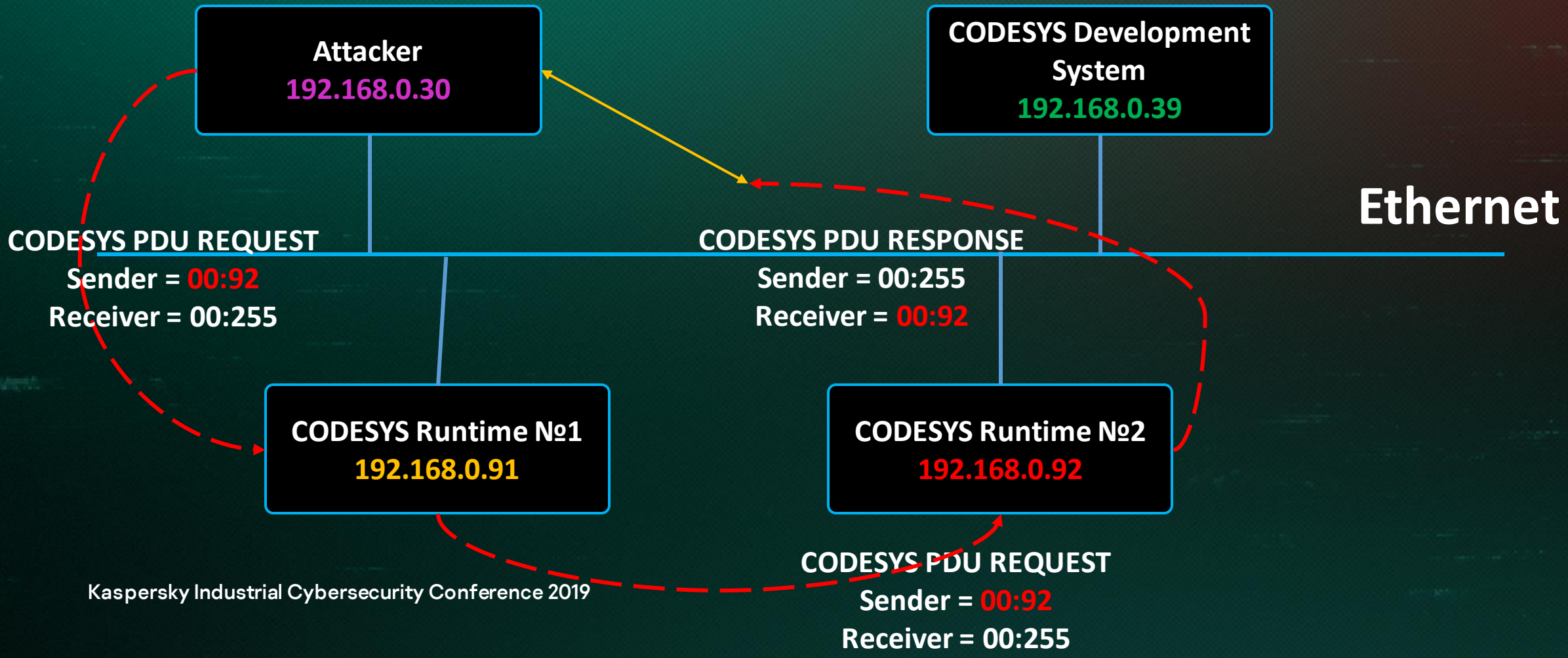
Color				
fields	Sender		Receiver	
	Port index	Address	Port index	Address
value	0 (1740)	0x5c (192.168.0.92)	0 (1740)	0xff (192.168.0.255)

CODESYS Runtime №1
192.168.0.91

192.168.0.92

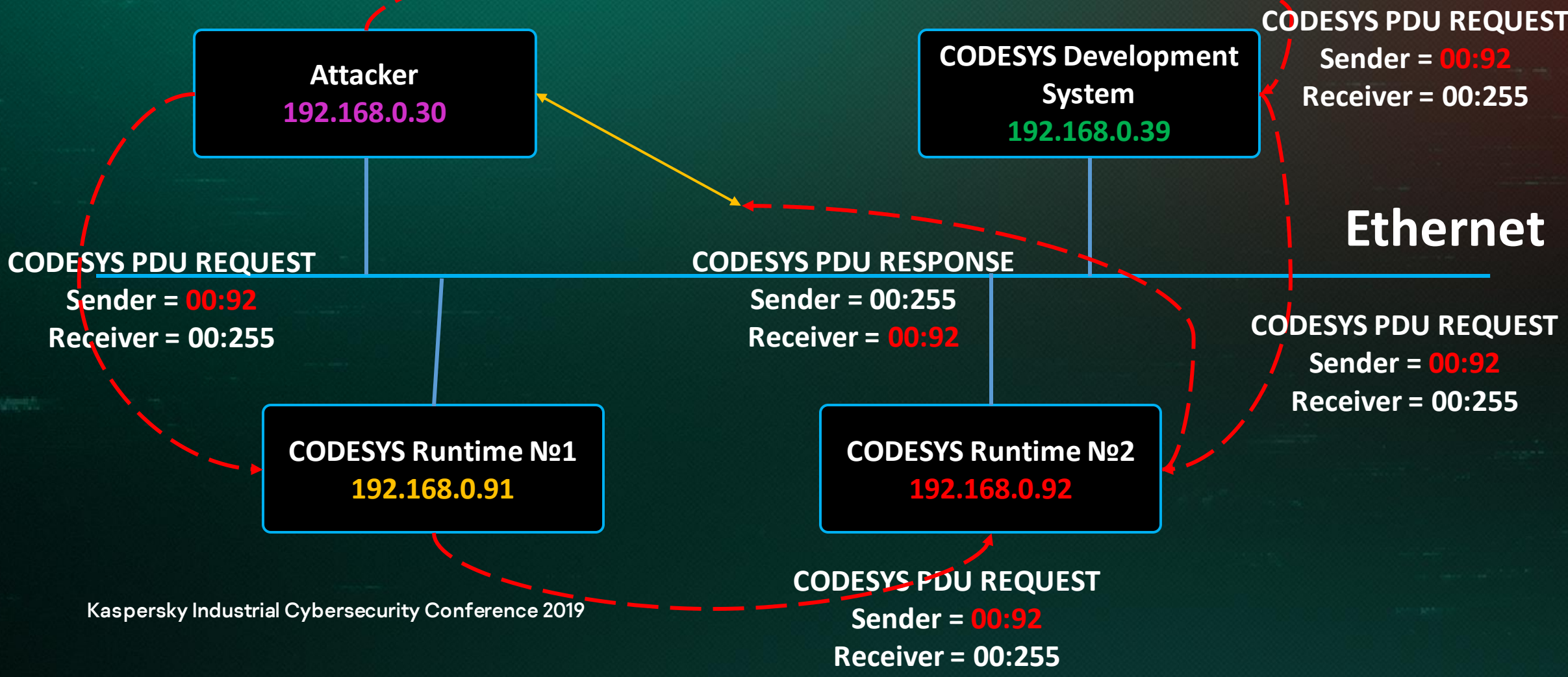
Vulnerability #1. Address spoofing

CODESYS address spoofing. Type #3 – With concealed receipt of a response to a request

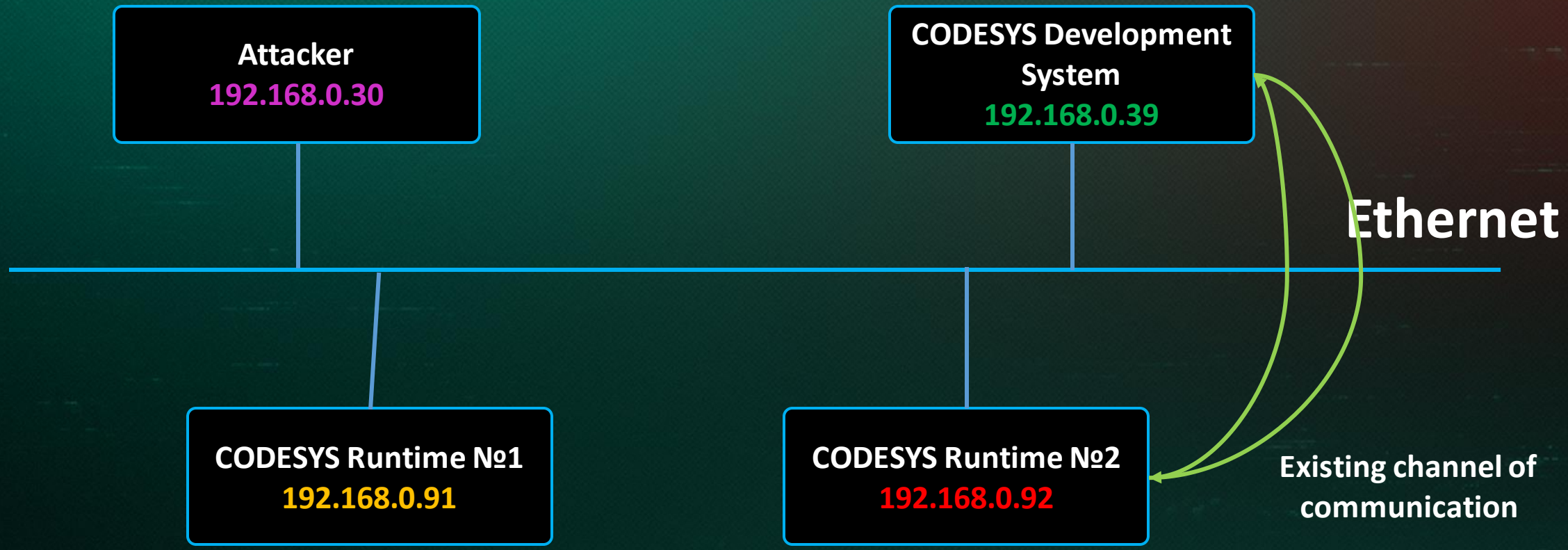


Vulnerability #1. Address spoofing

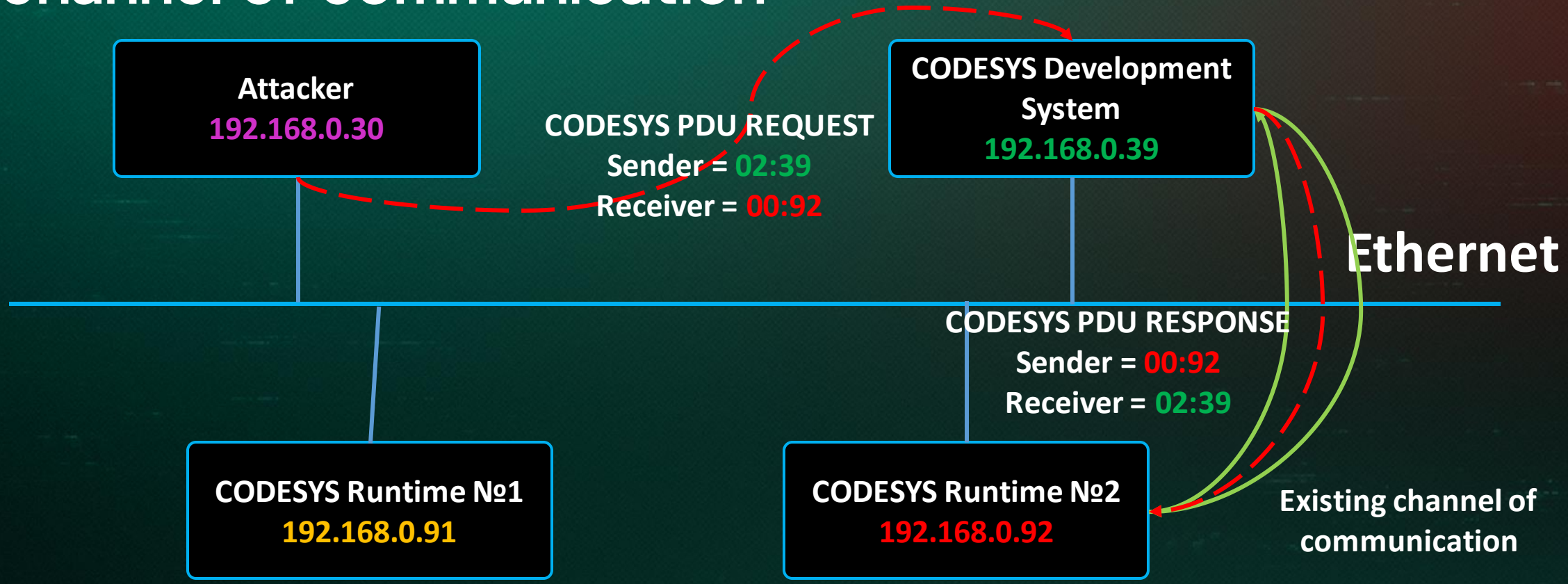
CODESYS address spoofing. Type #3 – With concealed receipt of a response to a request



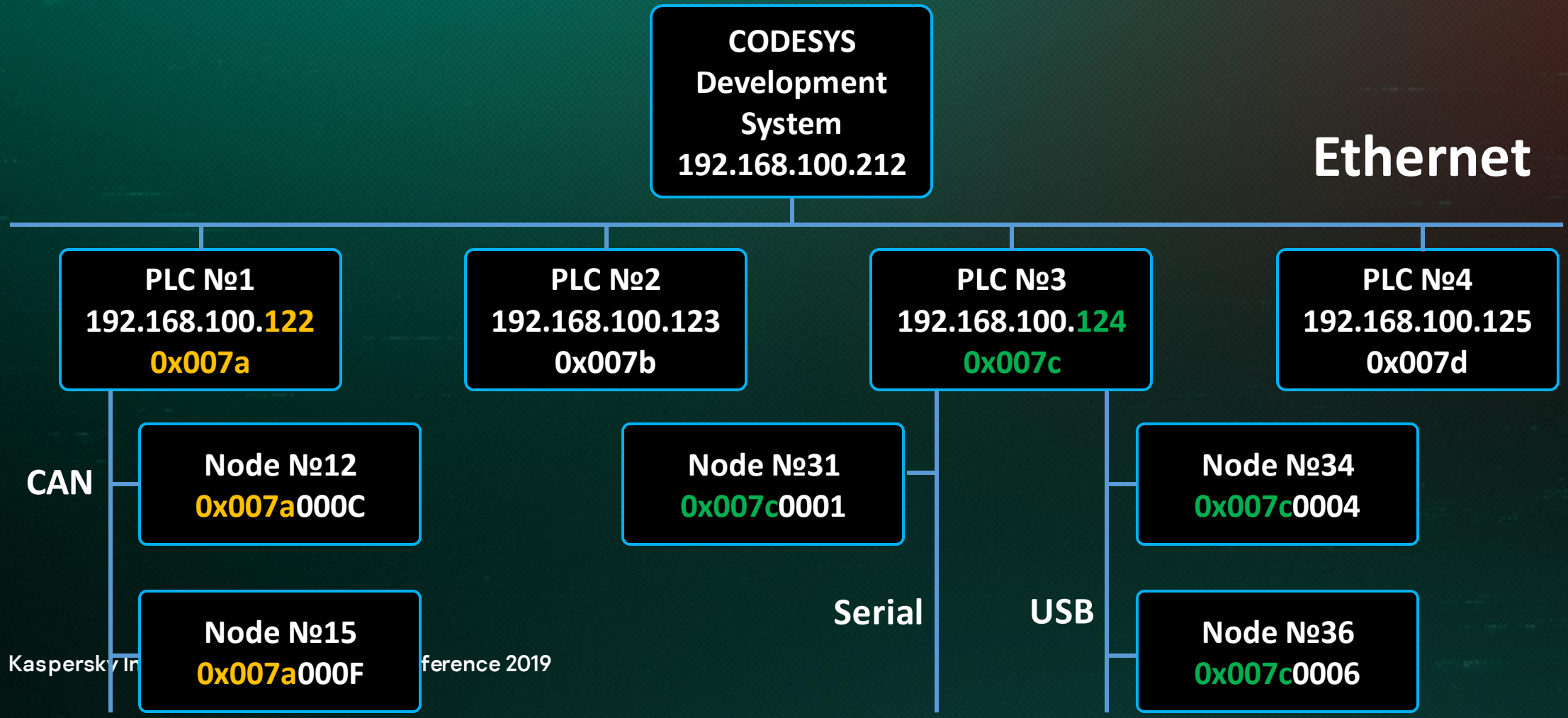
Vulnerability #2. Taking control of an existing channel of communication



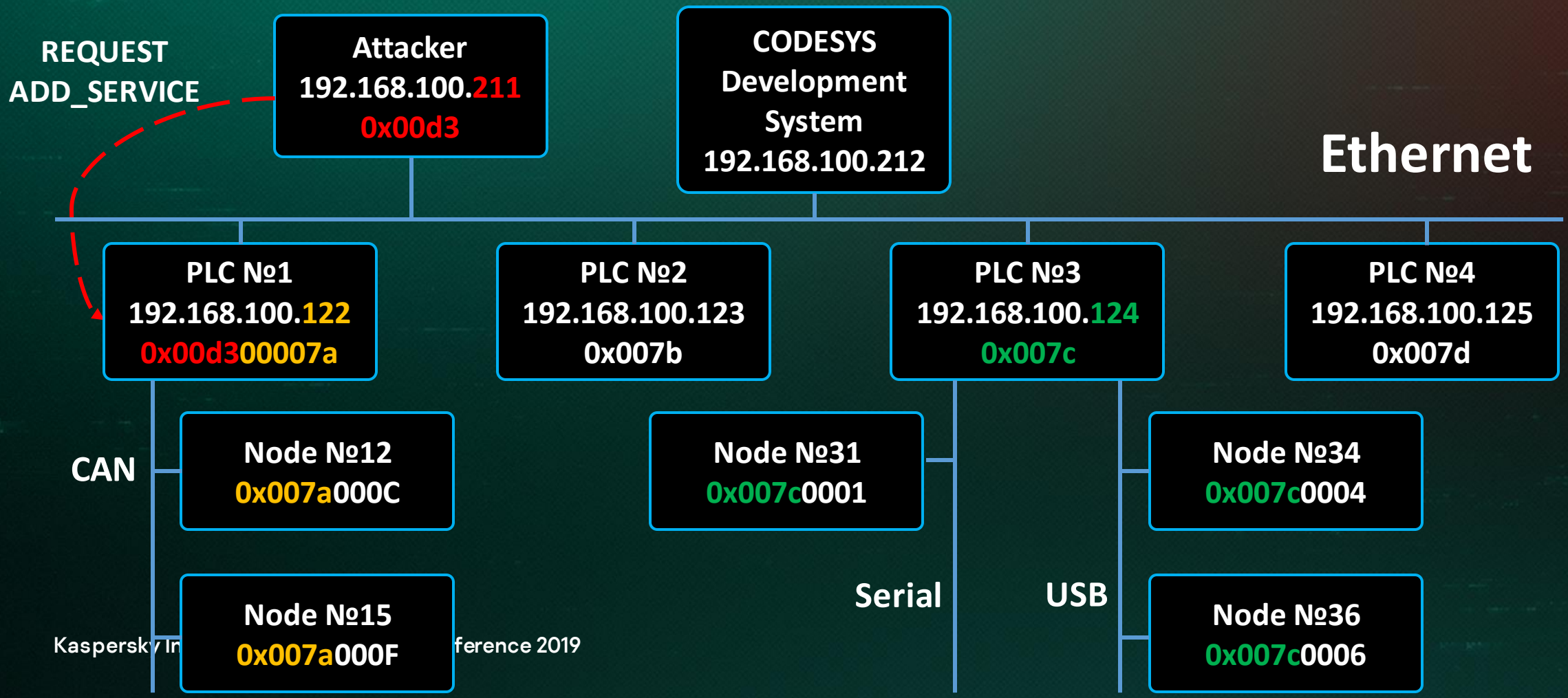
Vulnerability #2. Taking control of an existing channel of communication



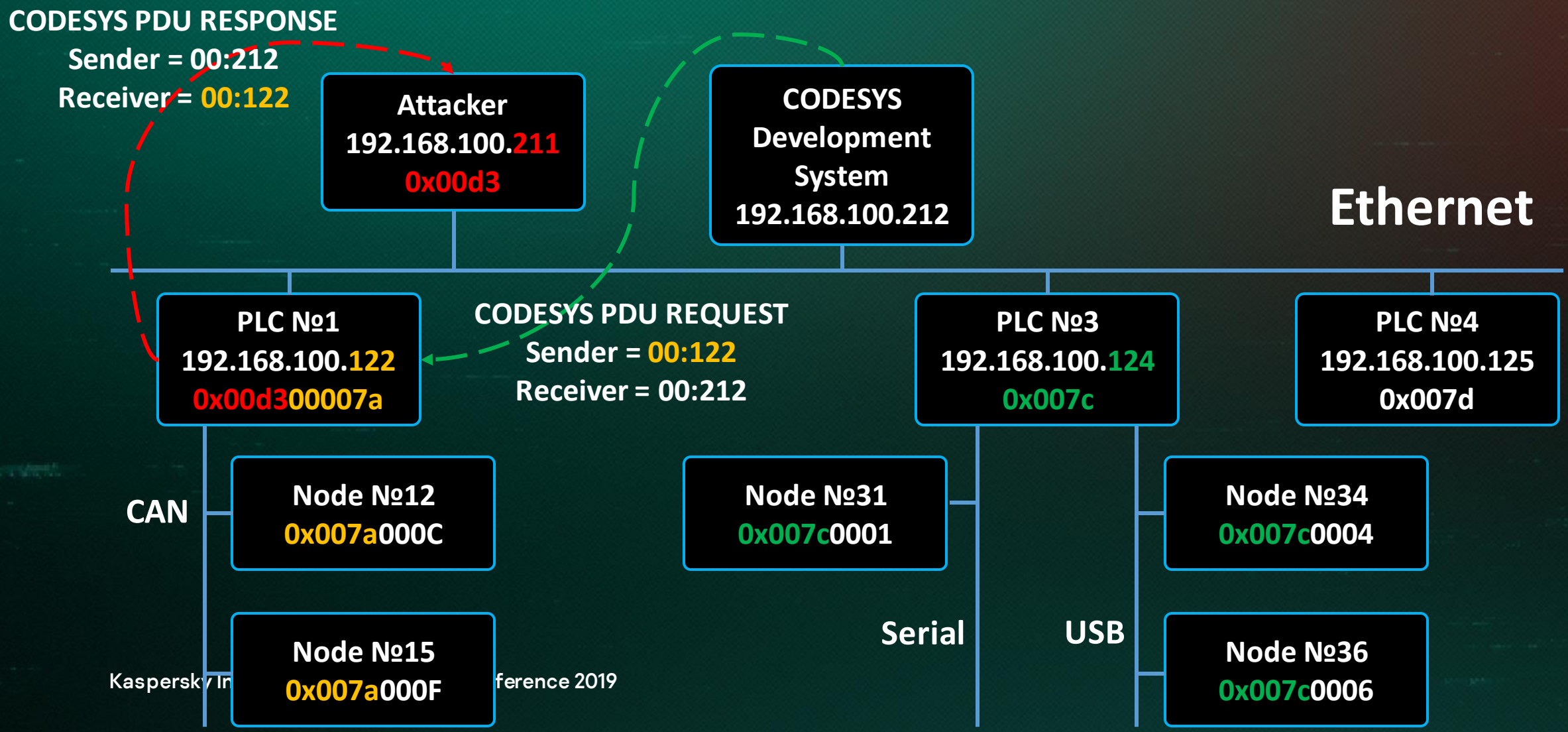
Vulnerability #3. MiTM



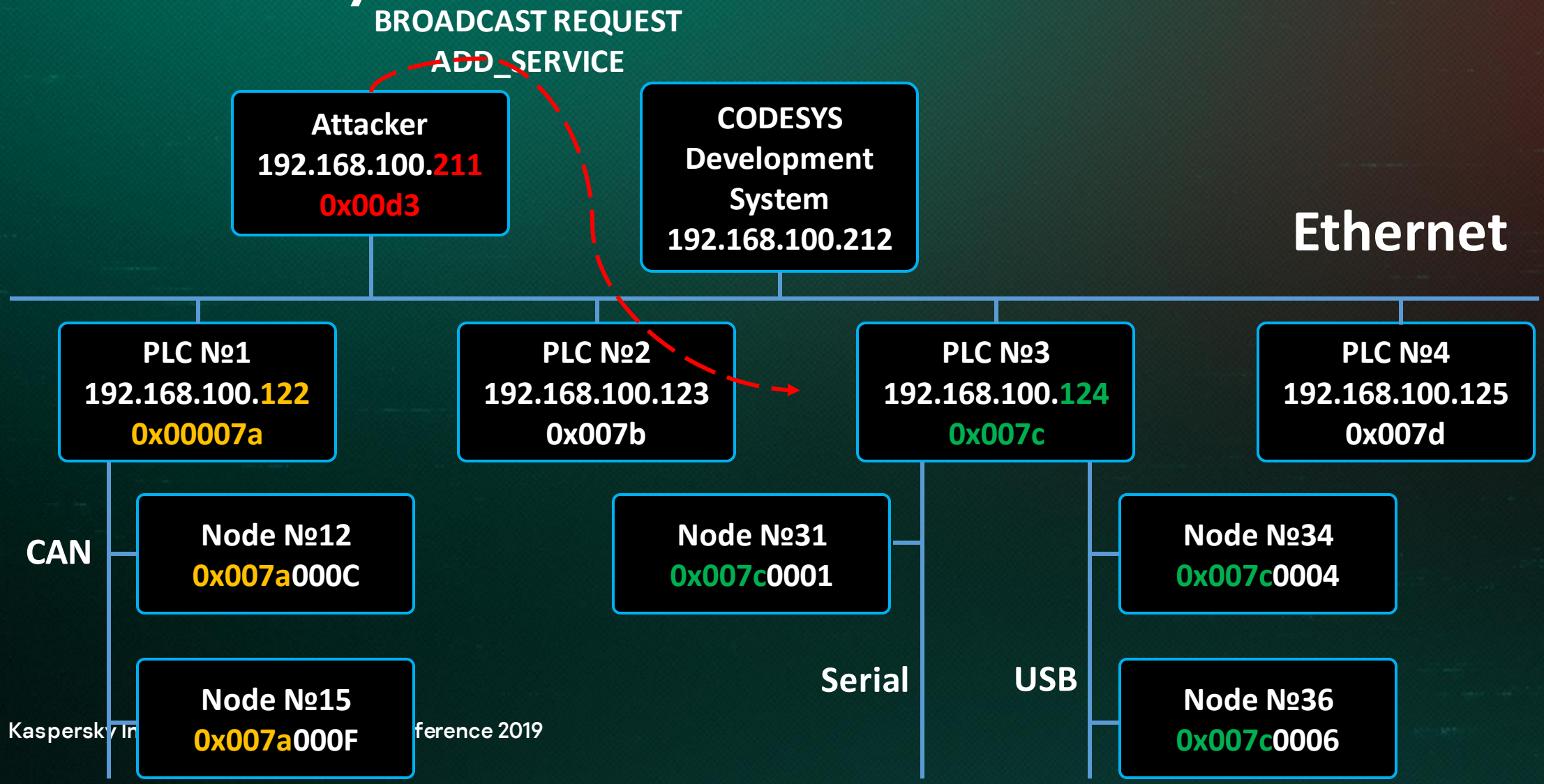
Vulnerability #3. MiTM



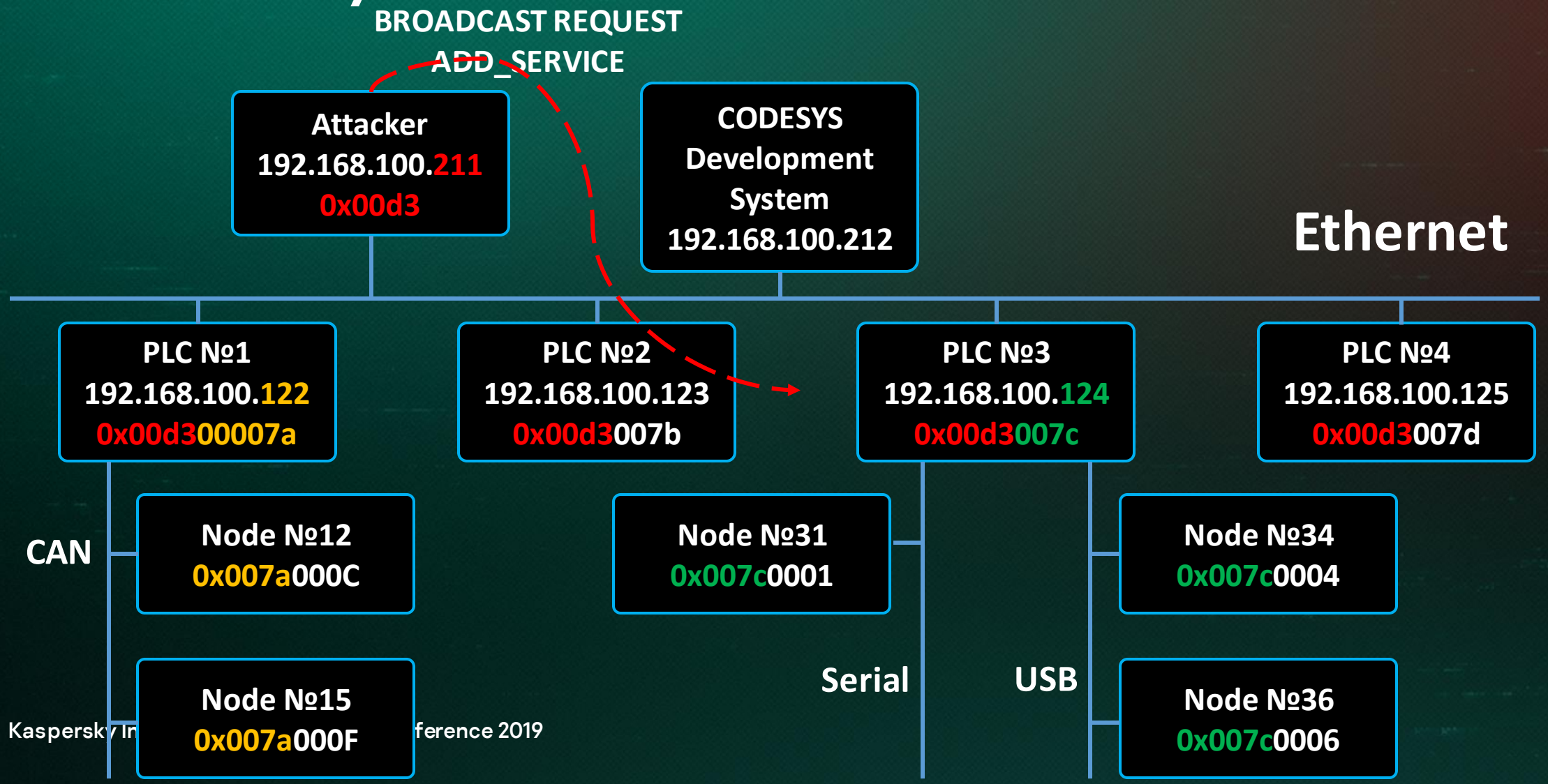
Vulnerability #3. MiTM



Vulnerability #3. MiTM



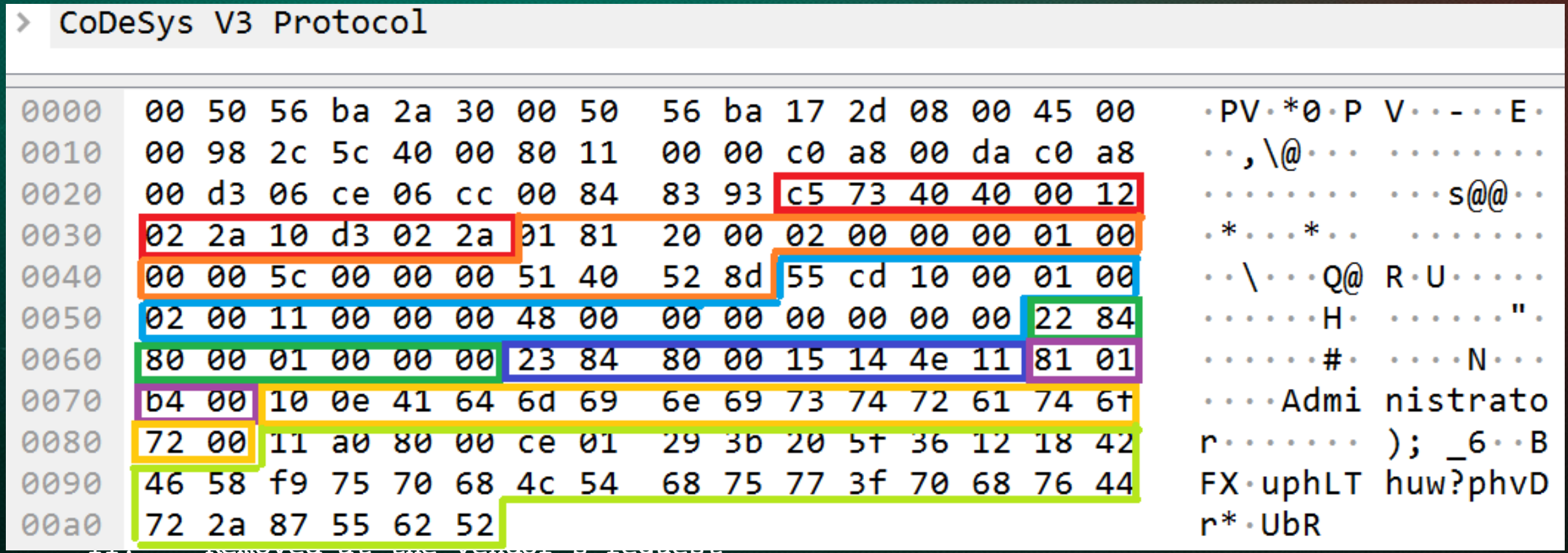
Vulnerability #3. MiTM



Vulnerability #4. Vulnerabilities in password encryption

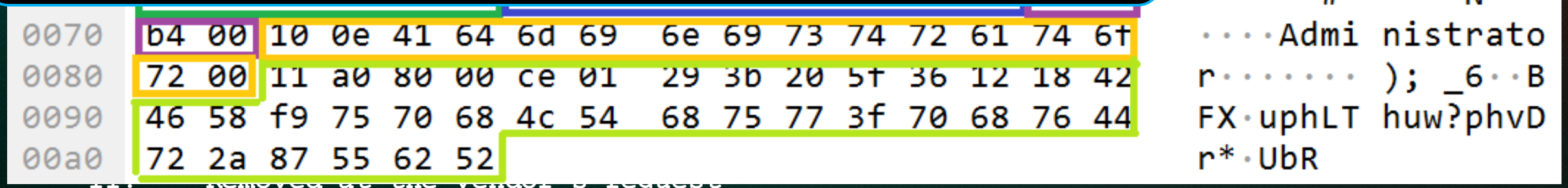
```
01: Removed at the vendor's request
02: {
03:  Removed at the vendor's request
04:  Removed at the vendor's request
05:  Removed at the vendor's request
06:  Removed at the vendor's request
07:  Removed at the vendor's request
08:  Removed at the vendor's request
09:  {
10:    Removed at the vendor's request
11:    Removed at the vendor's request
12:    Removed at the vendor's request
13:  }
14:  Removed at the vendor's request
15:}
```

CoDeSys V3 Protocol




```
12:     Removed at the vendor's request
13:   }
14:   Removed at the vendor's request
15:}
```

```
1: encrypted_password =
"\xce\x01\x29\x3b\x20\x5f\x36\x12\x18\x42\x46\x58\xf9\x75\x70
\x68\x4c\x54\x68\x75\x77\x3f\x70\x68\x76\x44\x72\x2a\x87\x55\
x62\x52"
2: KEY = "zeDR96EfU#27vuph7Thub?phaDr*rUbR"
3: for c, s in enumerate(encrypted_password):
4:     print chr(ord(KEY[c]) ^ ord(encrypted_password[c])),
5:
6: d m i  i s t M a t o
```




```
12:     Removed at the vendor's request
13:   }
14:   Removed at the vendor's request
15:}
```

Vulnerability #5. Vulnerability of application code

```
Header:
1: PROGRAM PLC_PRG
2: VAR
3:        magic: DWORD:= 16#DEADBEEF;
4: END_VAR

Body:
5: magic := magic + 16#BEEF;
```

Vulnerability #5. Vulnerability of application code

```
00c0  00 0a 0b 00 00 ea 6c 40  99 e5 08 50 a0 e3 04 50
00d0  85 e0 05 50 d9 e7 0a 00  55 e3 05 00 00 1a 01 40
00e0  a0 e3 0c 40 ca e5 6c 40  99 e5 01 40 84 e2 6c 40
00f0  89 e5 ff ff ff ea 30 42  bd e8 08 d0 8d e2 00 84
0100  bd e8 00 00 00 60 a0 01  c0 00 21 06 03 00 28 15
0110  01 00 22 b4 80 00 30 00  00 00 00 44 2d e9 0d a0
0120  a0 e1 30 00 2d e9 18 b0  9f e5 00 40 9b e5 0c 50
0130  9f e5 05 40 84 e0 00 40  8b e5 30 00 bd e8 00 84
0140  bd e8 ef be 00 00 70 38  00 00 a0 01 f0 08 21 06
0150  03 00 58 15 01 00 22 e4  88 00 60 04 00 00 00 44
0160  2d e9 0d a0 a0 e1 20 d0  4d e2 71 00 2d e9 00 40
0170  a0 e3 10 40 ca e5 00 40  a0 e3 20 40 0a e5 00 40
0180  a0 e3 1c 40 0a e5 00 40  a0 e3 18 40 0a e5 1f 40
0190  a0 e3 14 40 4a e5 1c 40  a0 e3 13 40 4a e5 1f 40
01a0  a0 e3 12 40 4a e5 1e 40  a0 e3 11 40 4a e5 1f 40
01b0  a0 e3 10 40 4a e5 1e 40  a0 e3 0f 40 4a e5 1f 40
01c0  a0 e3 0e 40 4a e5 1f 40  a0 e3 0d 40 4a e5 1e 40
01d0  a0 e3 0c 40 4a e5 1f 40  a0 e3 0b 40 4a e5 1e 40
01e0  a0 e3 0a 40 4a e5 1f 40  a0 e3 09 40 4a e5 0c 40
01f0  9a e5 00 50 94 e5 08 50  0a e5 10 d0 4d e2 08 40
0200  9a e5 00 40 8d e5 a8 43  9f e5 04 40 8d e5 9c b3
0210  9f e5 00 40 9b e5 0d 00  a0 e1 04 a0 2d e5 88 a3
0220  9f e5 04 a0 2d e5
```

```
00c0  00 00 60 4b 00 00 40 5f  00 00 00 00 00 60 a0 01
00d0  c8 00 21 06 03 00 e0 89  01 00 22 bc 80 00 38 00
00e0  00 00 00 44 2d e9 0d a0  a0 e1 08 d0 4d e2 10 08
00f0  2d e9 00 40 a0 e3 09 40  ca e5 00 40 a0 e3 08 40
0100  0a e5 00 40 a0 e3 04 40  4a e5 10 08 bd e8 08 d0
0110  8d e2 00 84 bd e8 00 00  00 60 a0 01 c8 00 21 06
0120  03 00 18 8a 01 00 22 bc  80 00 38 00 00 00 00 44
0130  2d e9 0d a0 a0 e1 08 d0  4d e2 10 08 2d e9 00 40
0140  a0 e3 09 40 ca e5 00 40  a0 e3 08 40 0a e5 00 40
0150  a0 e3 04 40 4a e5 10 08  bd e8 08 d0 8d e2 00 84
0160  bd e8 00 00 00 60 a0 01  d8 00 21 06 03 00 50 8a
0170  01 00 22 cc 80 00 48 00  00 00 00 44 2d e9 0d a0
0180  a0 e1 08 d0 4d e2 10 08  2d e9 00 40 a0 e3 09 40
0190  ca e5 00 40 a0 e3 08 40  0a e5 00 40 a0 e3 04 40
01a0  4a e5 14 40 9f e5 0c b0  9f e5 00 40 8b e5 10 08
01b0  bd e8 08 d0 8d e2 00 84  bd e8 70 38 00 00 ef be
01c0  ad de a0 01 d8 00 21 06  03 00 98 8a 01 00 22 cc
01d0  80 00 48 00 00 00 00 44  2d e9 0d a0 a0 e1 08 d0
01e0  4d e2 10 08 2d e9 00 40  a0 e3 09 40 ca e5 00 40
01f0  a0 e3 08 40 0a e5 00 40  a0 e3 04 40 4a e5 00 40
0200  a0 e3 0c b0 9f e5 00 40  8b e5 10 08 bd e8 08 d0
0210  8d e2 00 84 bd e8 7c 38  00 00 00 00 00 60 a0 01
0220  98 05 21 06 03 00
```

Vulnerability #5. Vulnerability of application code

```
01: 00 00 00 60      ANDVS       R0, R0, R0
02: A0 01 D8 00      SBCEQS      R0, R8, R0,LSR#3
03: 21 06 03 00      ANDEQ       R0, R3, R1,LSR#12
04: 50 8A 01 00      ANDEQ       R8, R1, R0,ASR R10
05: 22 CC 80 00      ADDEQ       R12, R0, R2,LSR#24
06: 48 00 00 00      ANDEQ       R0, R0, R8,ASR#32
07: 00 44 2D E9      STMFD       SP!, {R10,LR}
08: 0D A0 A0 E1      MOV         R10, SP
09: 08 D0 4D E2      SUB         SP, SP, #8
10: 10 08 2D E9      STMFD       SP!, {R4,R11}
11: 00 40 A0 E3      MOV         R4, #0
12: 09 40 CA E5      STRB        R4, [R10,#9]
13: 00 40 A0 E3      MOV         R4, #0
14: 08 40 0A E5      STR         R4, [R10,#-8]
15: 00 40 A0 E3      MOV         R4, #0
16: 04 40 4A E5      STRB        R4, [R10,#-4]
17: 14 40 9F E5      LDR         R4, =0xDEADBEEF; Write 0xDEADBEEF to R4
18: 0C B0 9F E5      LDR         R11, =0x3870   ; Write addr 0x3870 to R11
19: 00 40 8B E5      STR         R4, [R11]      ; Write 0xDEADBEEF to addr 0x3870
                                                   addr 0x3870
20: 10 08 BD E8      LDMFD       SP!, {R4,R11}
21: 08 D0 8D E2      ADD         SP, SP, #8
22: 00 84 BD E8      LDMFD       SP!, {R10,PC}
```

Vulnerability #5. Vulnerability of application code

```
01: 00 00 00 60        ANDVS       R0, R0, R0
02: A0 01 D8 00        SBCEQS      R0, R8, R0,LSR#3
03: 21 06 03 00        ANDEQ       R0, R3, R1,LSR#12
04: 50 8A 01 00        ANDEQ       R8, R1, R0,ASR R10
05: 22 CC 80 00        ADDEQ       R12, R0, R2,LSR#24
06: 48 00 00 00        ANDEQ       R0, R0, R8,ASR#32
07: 00 44 2D E9        STMFD       SP!, {R1
08: 0D A0 A0 E1        MOV         R10, SP
09: 08 D0 4D E2        SUB         SP, SP,
10: 10 08 2D E9        STMFD       SP!, {R4
11: 00 40 A0 E3        MOV         R4, #0
12: 09 40 CA E5        STRB        R4, [R10
13: 00 40 A0 E3        MOV         R4, #0
14: 08 40 0A E5        STR         R4, [R10
15: 00 40 A0 E3        MOV         R4, #0
16: 04 40 4A E5        STRB        R4, [R10,#-4]
17: 14 40 9F E5        LDR         R4, =0xDEADBEEF; Write 0xDEADBEEF to R4
18: 0C B0 9F E5        LDR         R11, =0x3870   ; Write addr 0x3870 to R11
19: 00 40 8B E5        STR         R4, [R11]      ; Write 0xDEADBEEF to addr 0x3870
                                                    addr 0x3870
20: 10 08 BD E8        LDMFD       SP!, {R4,R11}
21: 08 D0 8D E2        ADD         SP, SP, #8
22: 00 84 BD E8        LDMFD       SP!, {R10,PC}
```

```
Header:
1: PROGRAM PLC_PRG
2: VAR
3:         magic: DWORD:= 16#DEADBEEF;
4: END_VAR

Address magic is 0x3870
```

Vulnerability #5. Vulnerability of application code

```
01: 00 00 00 60      ANDVS       R0, R0, R0
02: A0 01 C0 00      SBCEQ       R0, R0, R0,LSR#3
03: 21 06 03 00      ANDEQ       R0, R3, R1,LSR#12
04: 28 15 01 00      ANDEQ       R1, R1, R8,LSR#10
05: 22 B4 80 00      ADDEQ       R11, R0, R2,LSR#8
06: 30 00 00 00      ANDEQ       R0, R0, R0,LSR R0
07: 00 44 2D E9      STMFD       SP!, {R10,LR}
08: 0D A0 A0 E1      MOV         R10, SP
09: 30 00 2D E9      STMFD       SP!, {R4,R5}
10: 18 B0 9F E5      LDR         R11, =0x3870    ; Write address 0x3870 to R11
11: 00 40 9B E5      LDR         R4, [R11]       ; Write contains of 0x3870 to R4
12: 0C 50 9F E5      LDR         R5, =0xBEEF     ; Write 0xBEEF to R5
13: 05 40 84 E0      ADD         R4, R4, R5      ; Add 0xBEEF to value in magic
14: 00 40 8B E5      STR         R4, [R11]       ; Write result to magic address
15: 30 00 BD E8      LDMFD       SP!, {R4,R5}
16: 00 84 BD E8      LDMFD       SP!, {R10,PC}
```

Vulnerability #5. Vulnerability of application code

```
01: 00 00 00 60        ANDVS       R0, R0, R0
02: A0 01 C0 00        SBCEQ       R0, R0, R0,LSR#3
03: 21 06 03 00        ANDEQ       R0, R3, R1,LSR#1
04: 28 15 01 00        ANDEQ       R1, R1, R8,LSR#1
05: 22 B4 80 00        ADDEQ       R11, R0, R2,LSR#
06: 30 00 00 00        ANDEQ       R0, R0, R0,LSR R
07: 00 44 2D E9        STMFD       SP!, {R10,LR}
08: 0D A0 A0 E1        MOV         R10, SP
09: 30 00 2D E9        STMFD       SP!, {R4,R5}
10: 18 B0 9F E5        LDR         R11, =0x3870    ; Write address 0x3870 to R11
11: 00 40 9B E5        LDR         R4, [R11]       ; Write contains of 0x3870 to R4
12: 0C 50 9F E5        LDR         R5, =0xBEEF     ; Write 0xBEEF to R5
13: 05 40 84 E0        ADD         R4, R4, R5      ; Add 0xBEEF to value in magic
14: 00 40 8B E5        STR         R4, [R11]       ; Write result to magic address
15: 30 00 BD E8        LDMFD       SP!, {R4,R5}
16: 00 84 BD E8        LDMFD       SP!, {R10,PC}
```

```
Body:
5: magic := magic + 16#BEEF;

Address magic is 0x3870
```

In conclusion

- All discovered vulnerabilities could have been found by the community if the protocol specification had been available
- Security by obscurity approach is not the best strategy for protecting

Kaspersky Industrial Cybersecurity Conference 2019

September 18-20, 2019, Sochi, Russia

kaspersky

Thank you!

```
pi@raspberrypi:~ $ ./opt/codesys/bin/codesyscontrol.bin -vvvvvvv
CODESYS Control V3.5.12.0 for ARM - build Dec 18 2017
type:4102 id:0x00000010 name:CODESYS Control for Raspberry Pi SL
vendor: 3S - Smart Software Solutions GmbH
buildinformation: <none>

 _________
< ... bye >
 ---------
        \   ^__^
         \  (--)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

kaspersky.com

ics-cert.kaspersky.com