DEVELOPER DAYS 2014 EUROPE

Security aspects of feature rich, connected embedded devices

Till Adam - KDAB
Nicolas Mayencourt - Dreamlab

Who are we, and why are we on this stage?

Till Adam

- Responsible for services at KDAB
- Veteran of Qt consultancy
- Helping people build devices

Nicolas Mayencourt

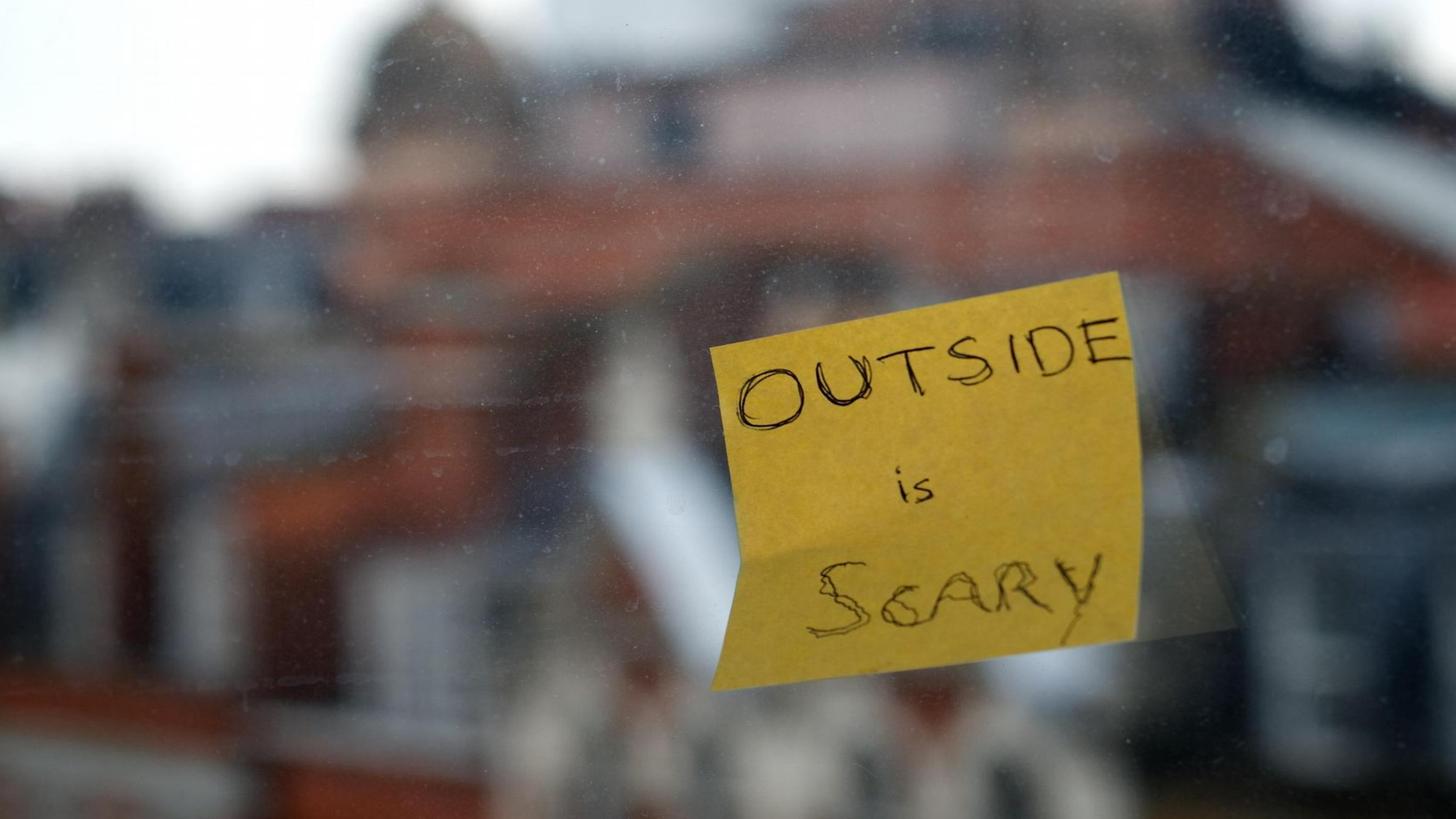
- Founder and CEO of Dreamlab
- Veteran of security research and consultancy
- Helping people secure devices

The Big Picture

The Internet of Things

The Mobile Age

Commoditization


OUTSIDE
is
SCARY

The Internet of Things

Nearly everything is expected to be connected to the internet.

Your thermostat, your toothbrush, your TV, your blood pressure monitor and your washing machine might already be.

Whatever you build as your job, chances are it will be connected very soon, if it is not already.

The Mobile Age

Everybody has a smart phone.

The user experience expectation is dominated by mobile devices and gaming consoles, not PCs or kitchen appliances.

Touch screens are becoming ubiquitous.

Instant access from anywhere and any device is the norm.


0
1
2
3
4
5
6
Kg /cm²

Commoditization

Cost pressure is increasing everywhere.

At the same time feature expectations increase.

More and more expensive hardware and software is needed.

As a result more and more commodity hardware and software is used to make devices.

Like it or not ...

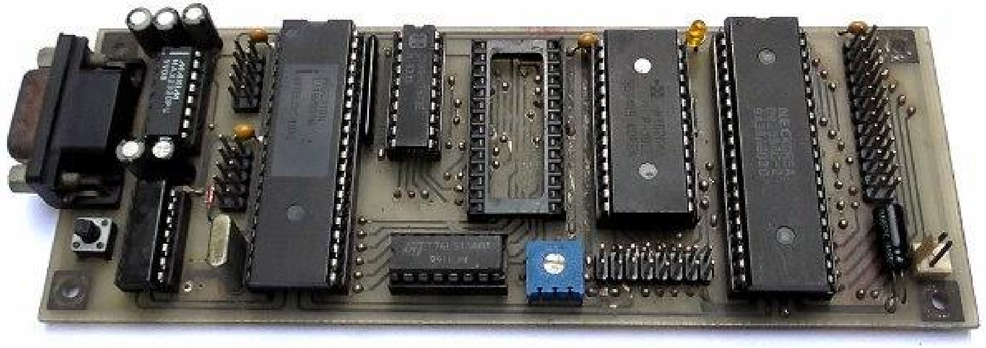
Embedded devices are :

- **becoming connected**
- **becoming very complex**
- **need interfaces “like the iPhone”**
- **use commodity boards, periphery, operating systems and tools**

Changing Landscape of Embedded Devices

The Past

- Specialized hardware
  - Expensive
  - Integrated circuits
- Single process
- Optimized code (C/ASM/...)
- Update only via HW (EEPROM)
- No / limited connectivity
- Specialized solution / sw



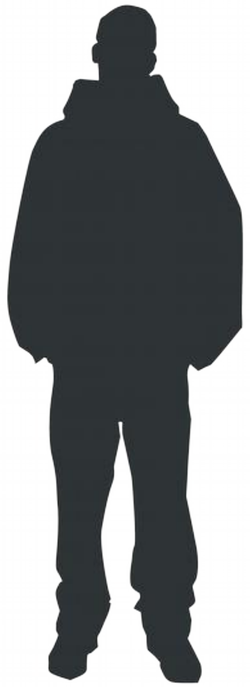
Today

- Commodity Hardware
  - Full-blown Computer (SoC)
  - Cheap
  - Massive interface support
  - Connectivity (IP/Wireless/...)
- Internet visibility
- Complex Software-Stack
  - Need for updates
- ... more like a Computer-Appliance

Security Assumptions Change

- Embedded Systems are black boxes
- … can't be analyzed without extensive knowledge and funds
- How should one find our «secret backdoor» if even we have trouble using it?
- Endusers will always stay up to date with our newest release
- … because they know how to update
- … read our newsletter regularly
- No one will connect this device directly to the Internet

Why does this matter ?
I am not a target.



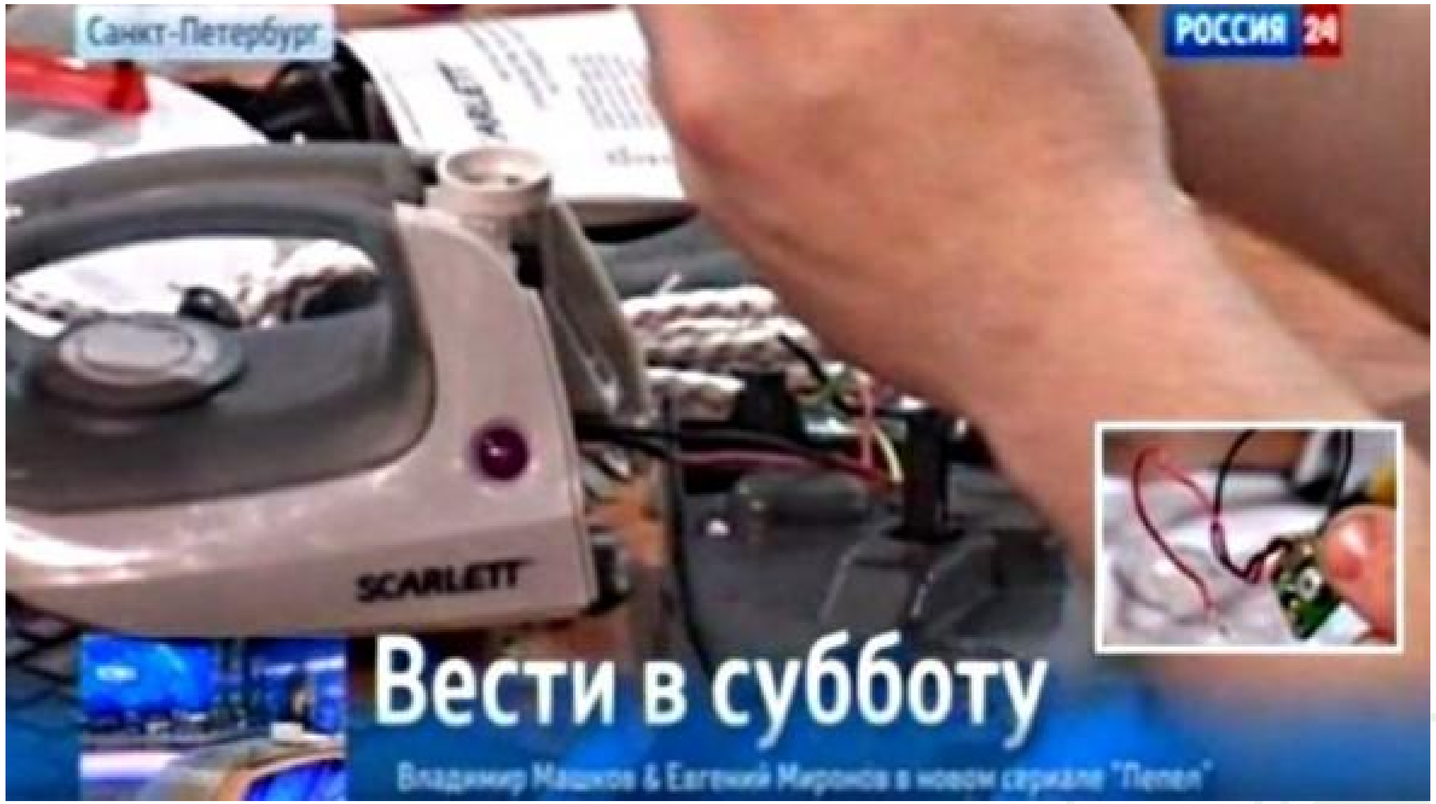
IOT & commoditization is a gamechanger

Really ?
Electronic crime is different

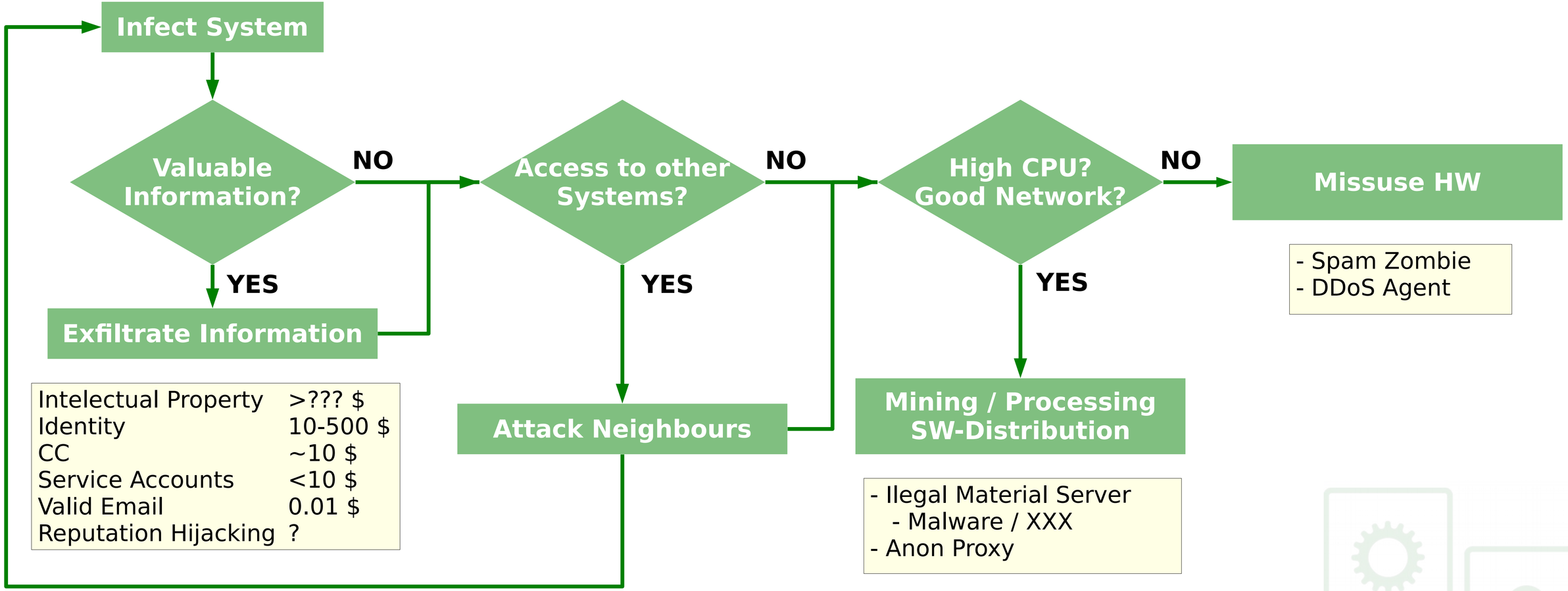
Something to protect ?

Embedded: Iron

- Chinese appliances are shipping with malware-distributing WiFi chips
- 29. Oct. 2013

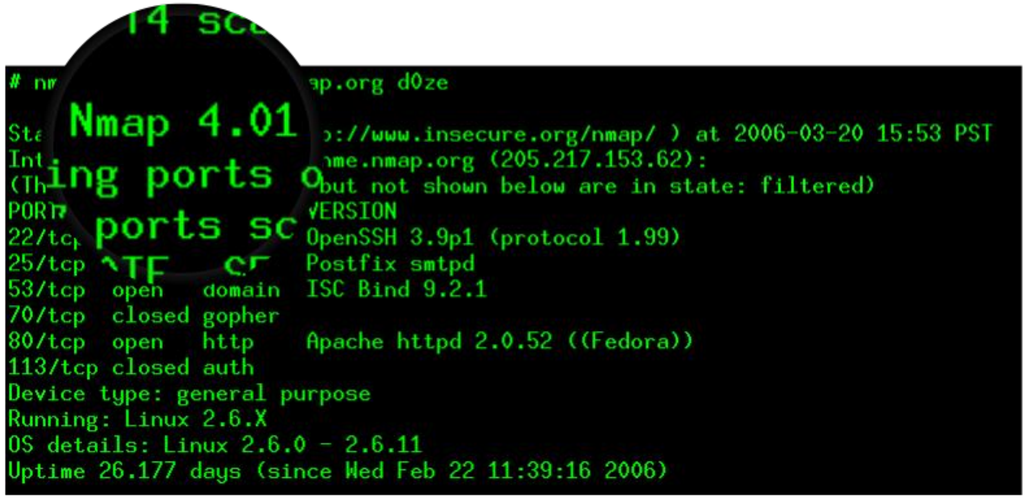


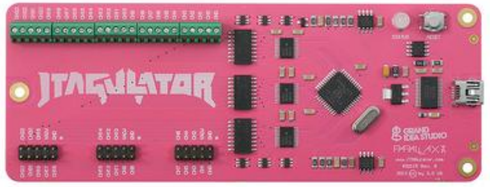
I'm not a target?



Available & cheap tools

- **Standard IT-Security tools**
- LogicAnalyzer
- BusPirate / JTAGULATOR
- USB-Oszilloscope
- USB-Microscope
- Public chip specifications
- Free compiler toolchains
- Cheap development boards

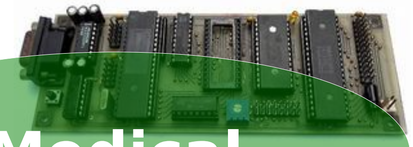




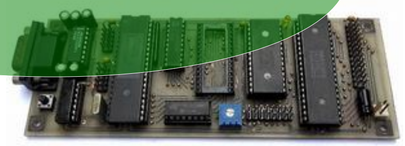
Past: Specialized Platforms & Attacks

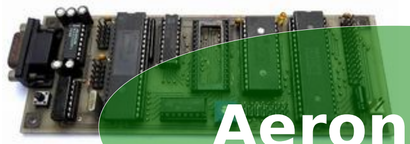
Automotive

Medical



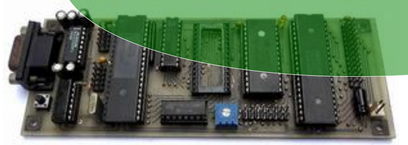
Consumer





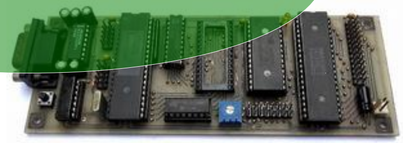
Aeronautic

Industrial



Military

Finance

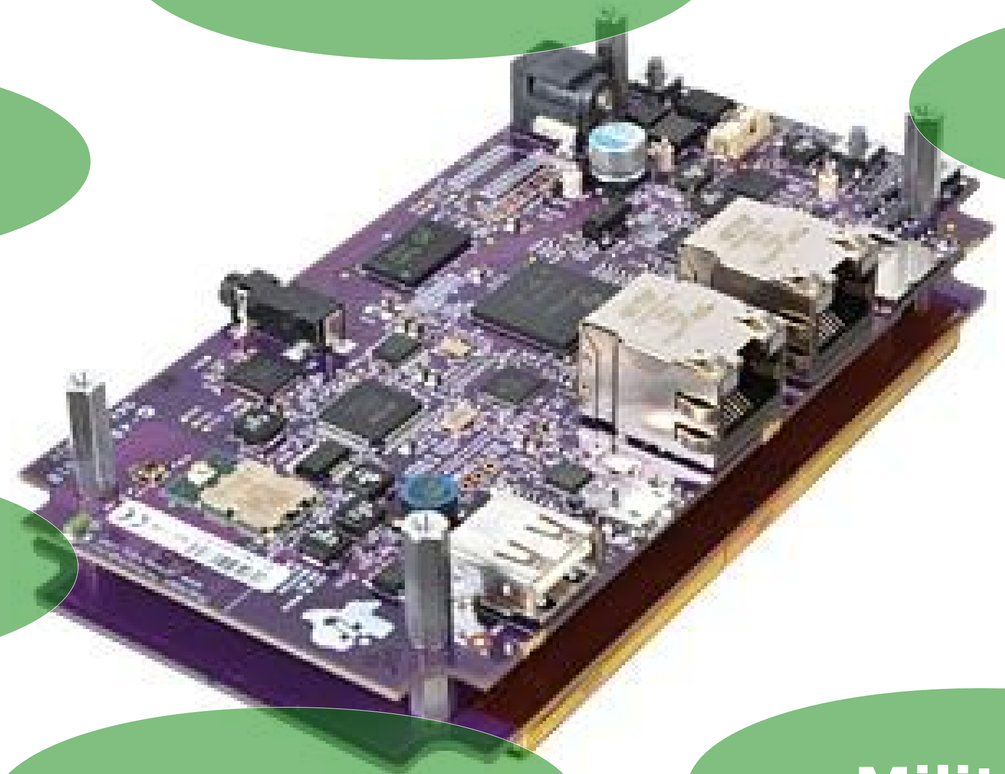


Today: Commoditization
One exploit fits all

Automotive

Consumer

Medical



Aeronautic

Industrial

Finance

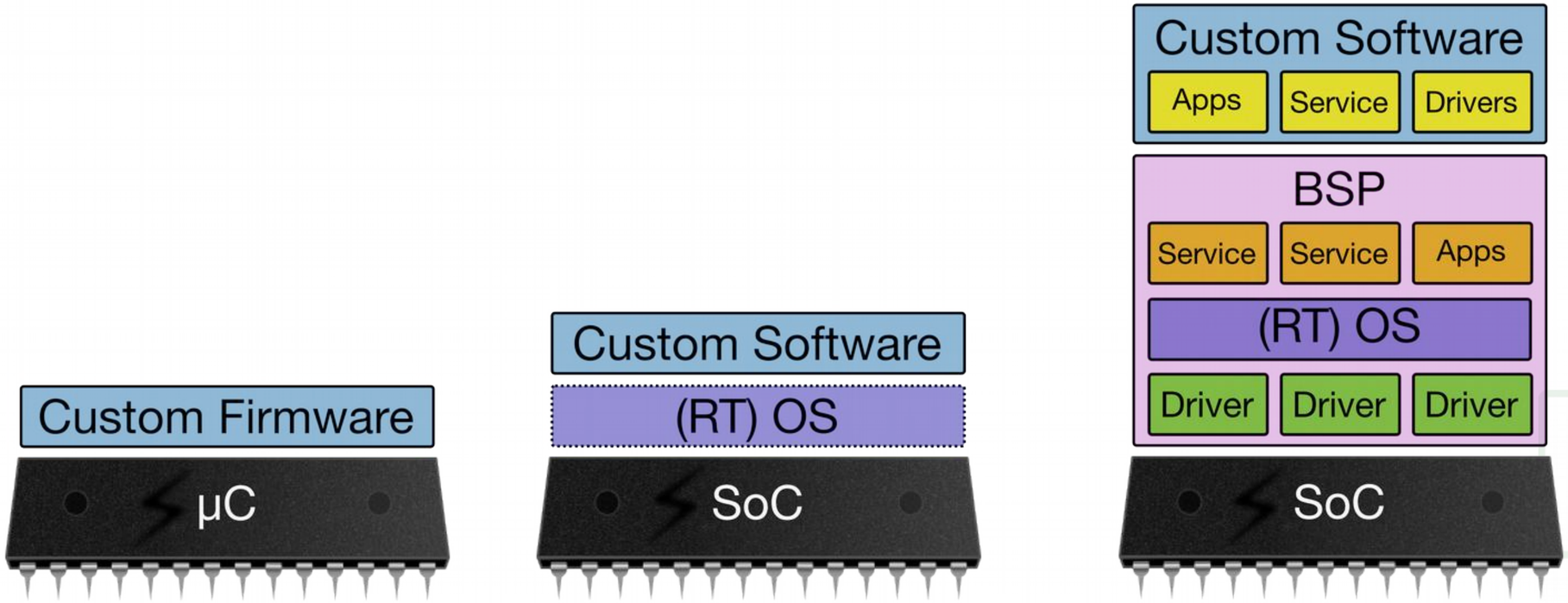
Military

Connectivity

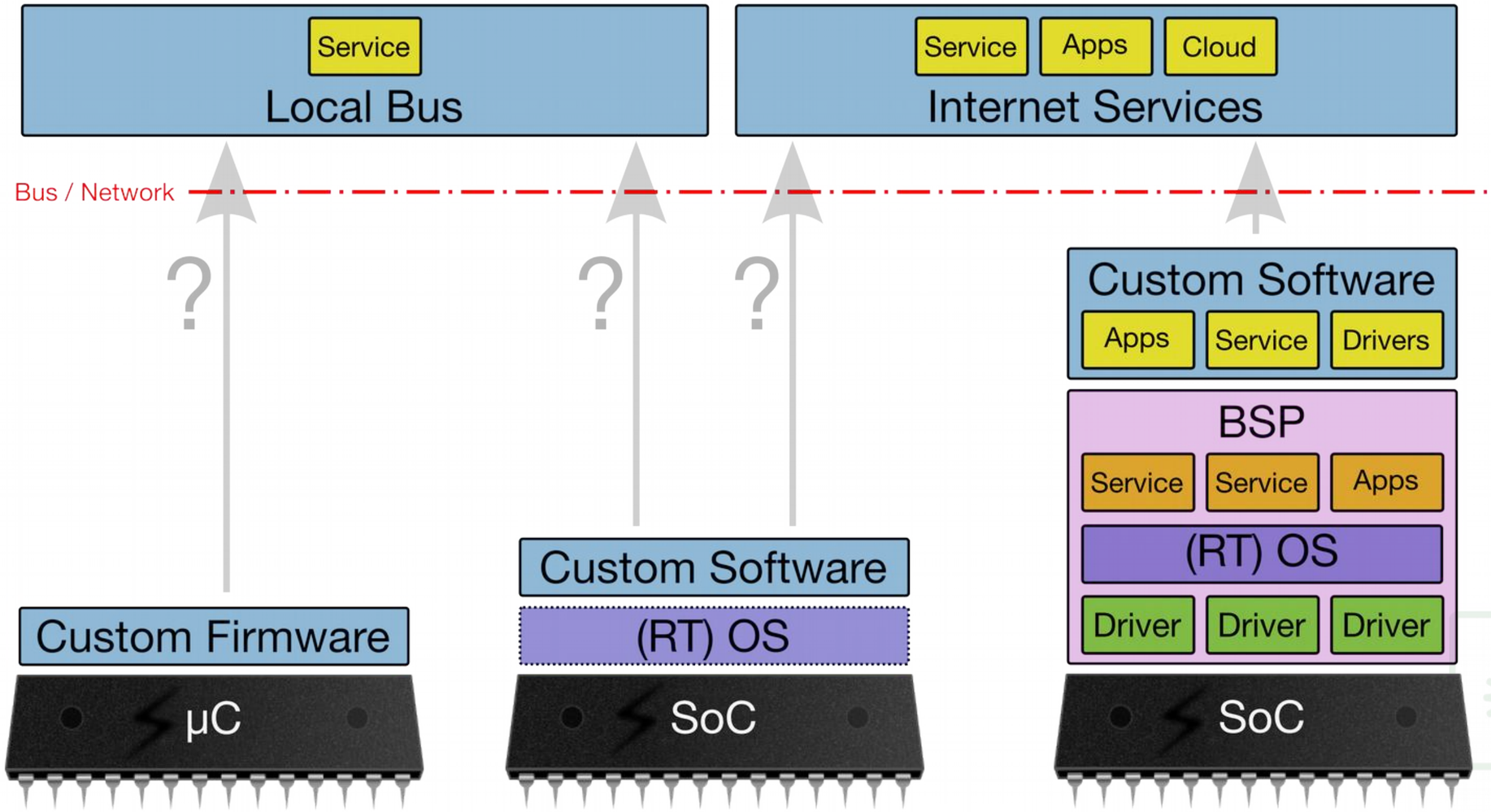
GSM

- GSM
- UMTS
- LTE
- WLAN
- Bluetooth
- GPS
- USB

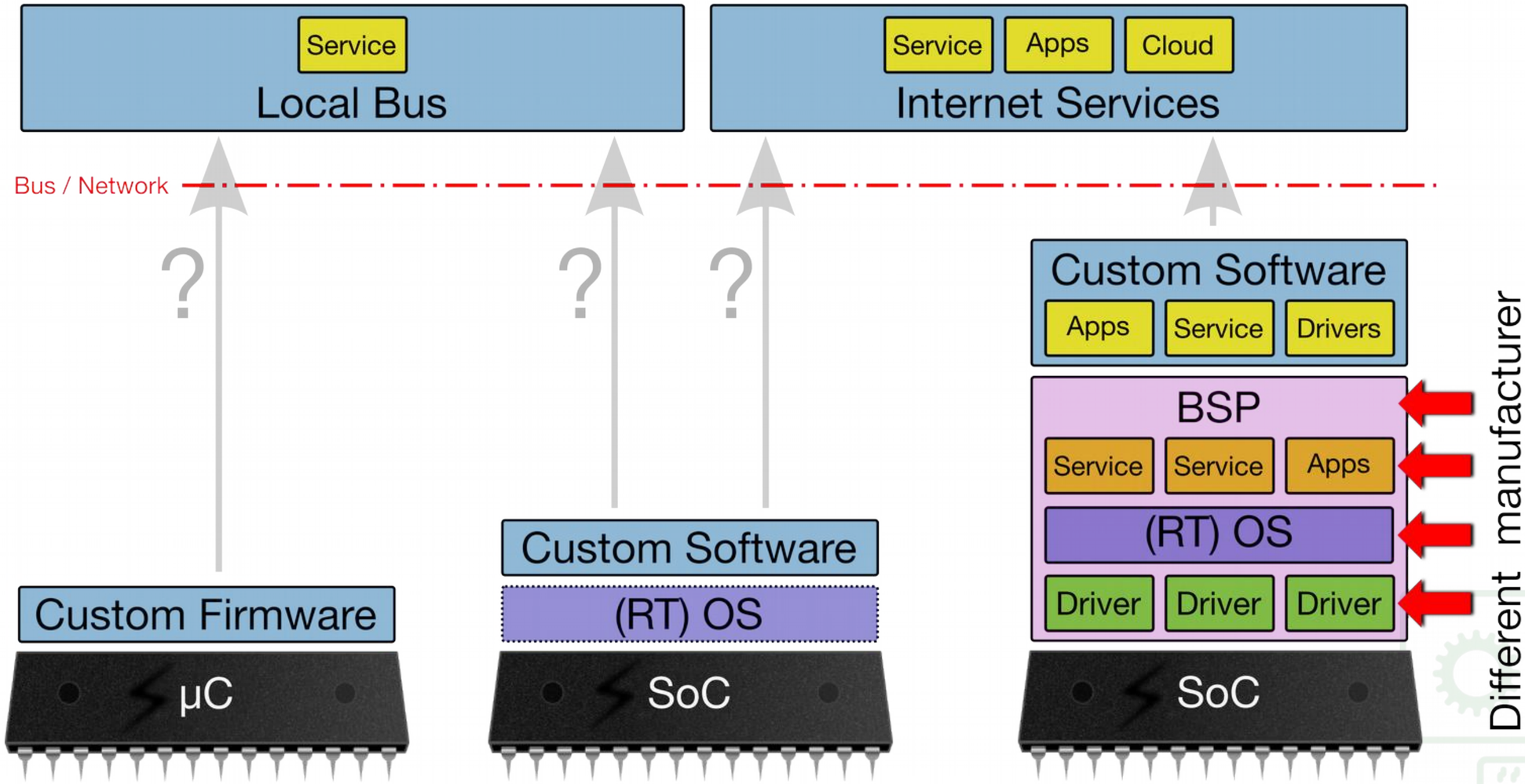
Software / Hardware Stack



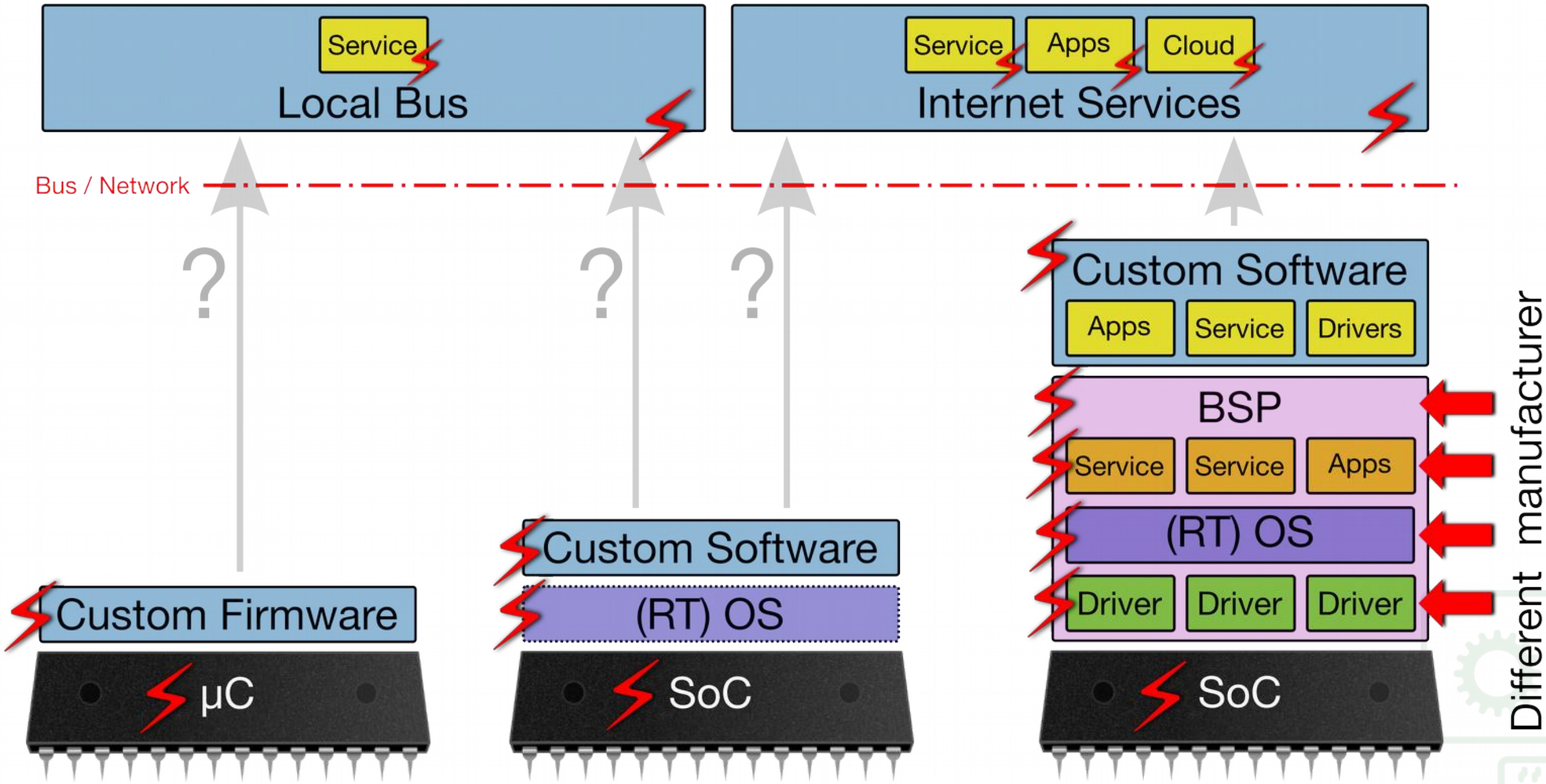
Software / Hardware Stack



Software / Hardware Stack



Software / Hardware Stack



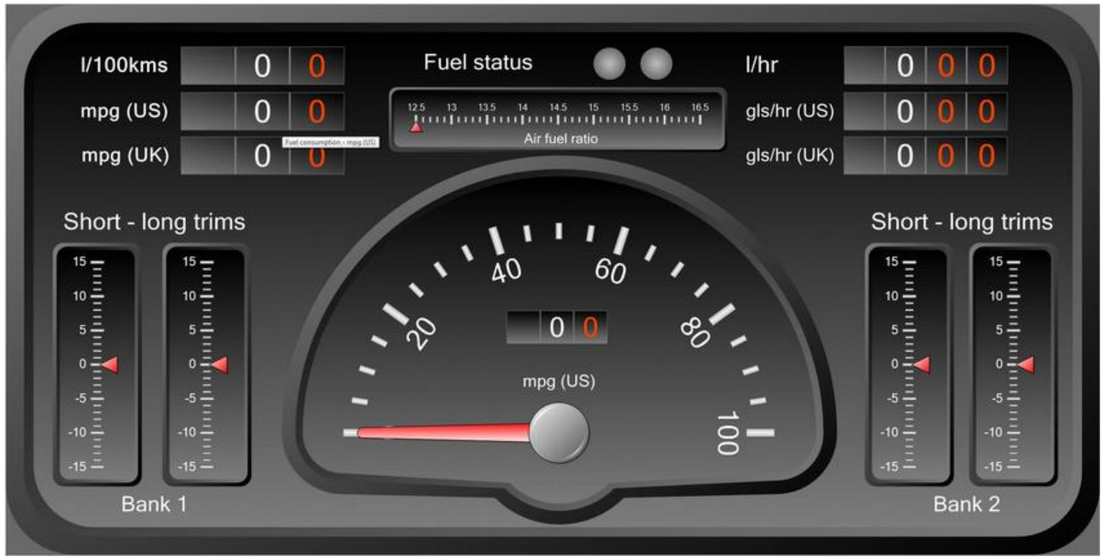
BSP's


```php
<!-- execute-command.php -->
<?php
    $command = $_GET['command'];
    $output = shell_exec($command);

    echo $output
?>
```

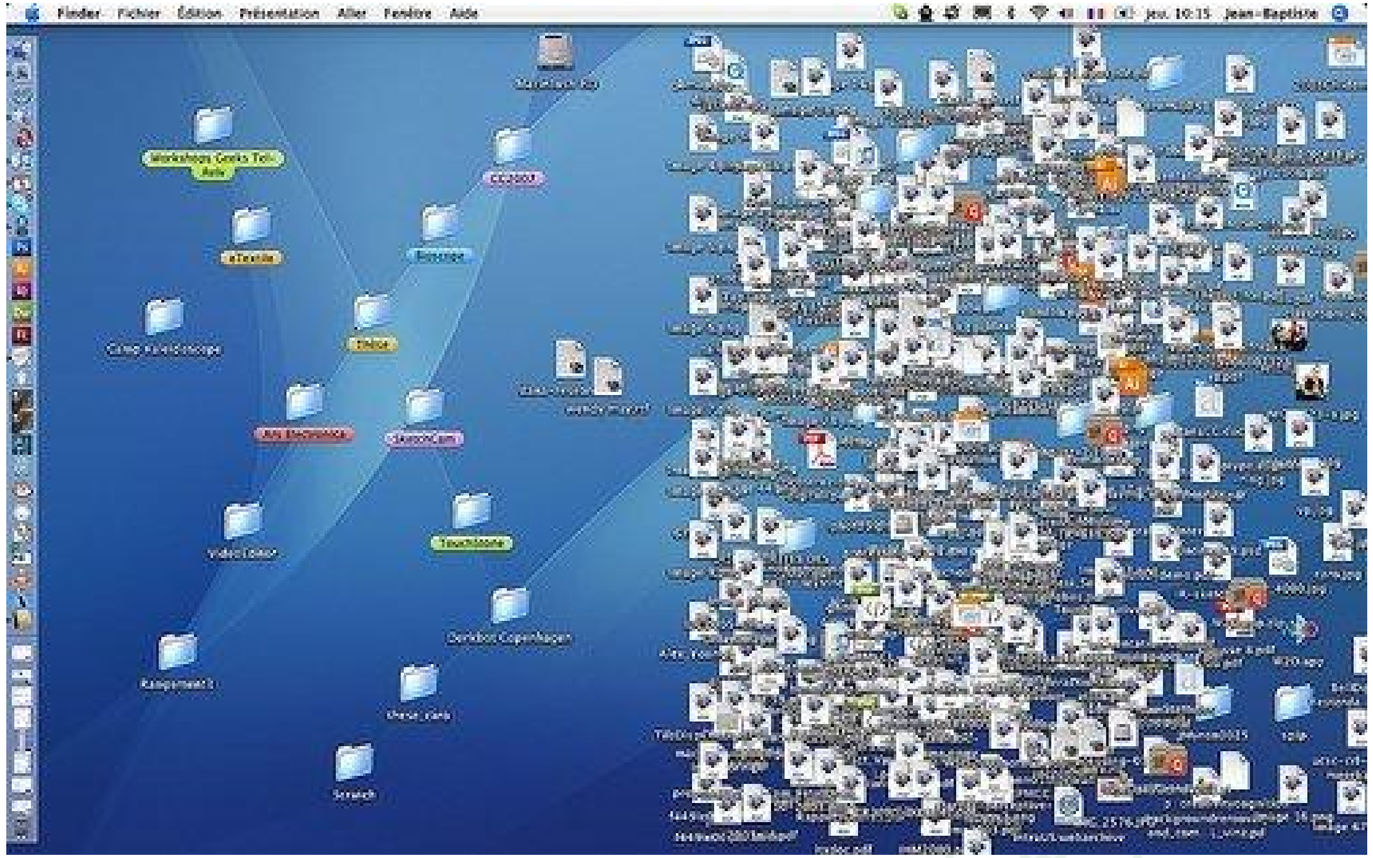
Disclaimer: TI clearly states that this is not meant / fit for production

Disclaimer disclaimer: There's nothing wrong with the BSP form TI !

What you see != What you get



© http://mikesshop.net/

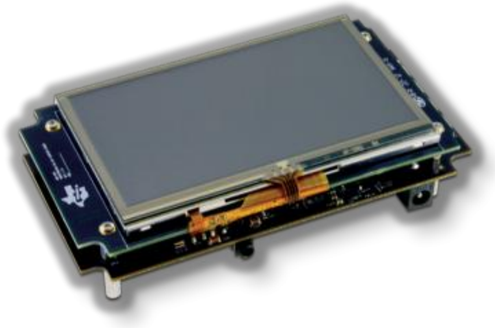


© http://www.apartmenttherapy.com/

What you see != What you get

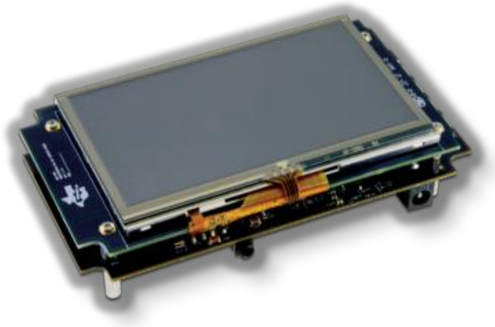
- Leftovers on Disk / Firmware-Images
  - .bash_history (typos / internal staging servers...)
  - Log-files
  - Logos / documents from different customer
  - SSH-keys / credentials / SSL-Certificates
  - Demo-Software (from BSP's)
  - Unneeded services
  - Development leftovers (debug / backdoors)
- Outdated software / Backup-Files
- Deleted files (still visible)
- License violations
- We are not yet even talking serious security here

Modern Software Protection



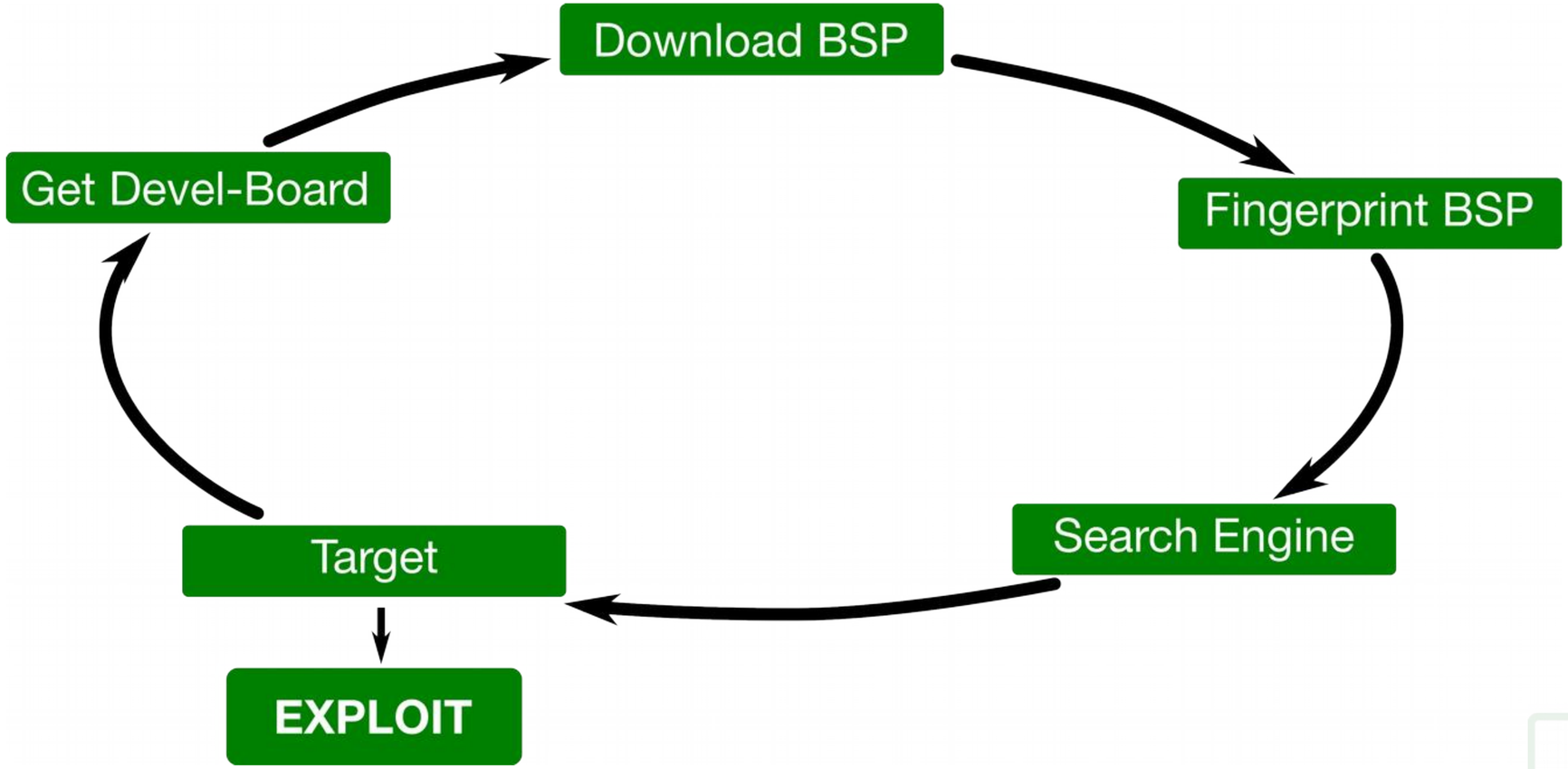
Against existence of Exploits		
Safe programming / secure coding	✓	?
Input validation	✓	?
Static / dynamic code analysis	✓	?
Against exploitation		
ASCII armored address space	✓	?
Stack guard / stack protection (Canaries)	✓	?
No executable stack (DEP)	✓	?
Address spcace layout randomization (ASLR)	✓	?

Modern Software Protection



Exploit prevention / detection		
Antivirus	✓	?
Hostbased Intrusion Detection Systems	✓	?
Intrusion Prevention System	✓	?
Software maintenance		
Daily Updates	✓	?
Encrypted Communication Channels	✓	?
Signed Software	✓	?

Where does it lead to?


Download BSP
Fingerprint BSP
Search Engine
Target
EXPLOIT
Get Devel-Board

Get Development Device

Texas Instruments AM335x Starter Kit

Fingerprint BSP

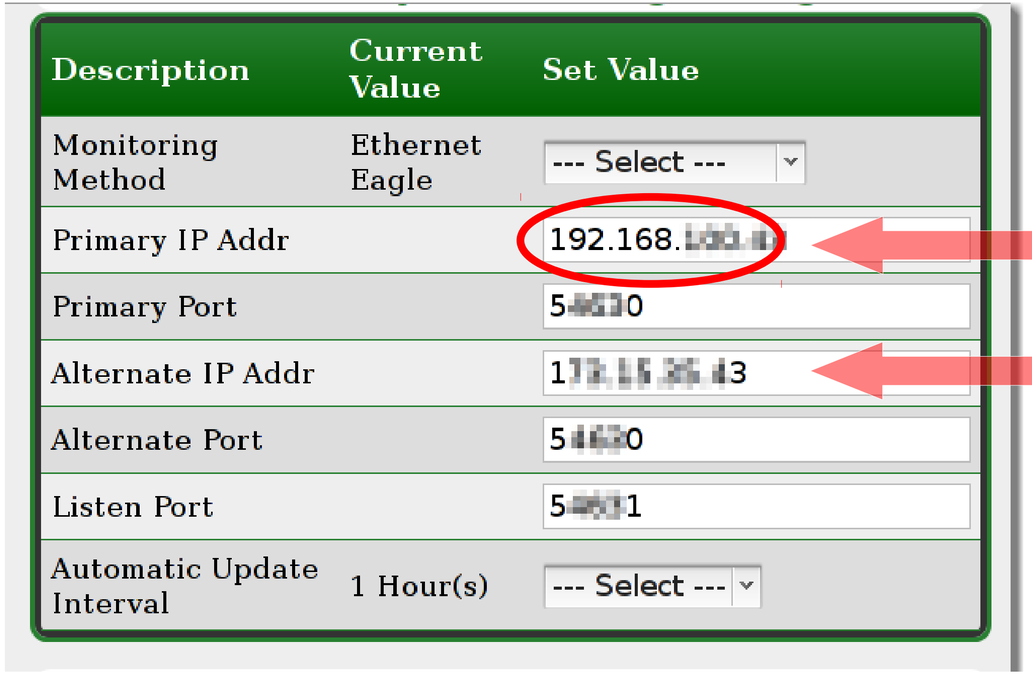
- Comes with openembedded „Arago 2013.12 – am335x-evm“
- Busybox Linux (Kernel 3.1)
- Telnet
- Dropbear sshd 2012.55
- lighttpd 1.4.32 (on port 80)
- thttpd 2.25b 29dec2003 (on port 8080)
- Only root user with default password „root“
- Matrix GUI (HTML5 / PHP Webapplication)

Search for similar Devices

```
# Nmap 6.47 scan
Nmap scan report for productive-device
Host is up, received user-set

PORT     STATE          SERVICE      REASON       VERSION
21/tcp   open           ftp          syn-ack      vsftpd 2.2.2
22/tcp   open           ssh          syn-ack      Dropbear sshd 0.51 (protocol 2.0)
23/tcp   open           telnet?      syn-ack
80/tcp   open           http         syn-ack      lighttpd 1.4.33

161/udp  open           snmp         udp-response SNMPv1 server (public)
```

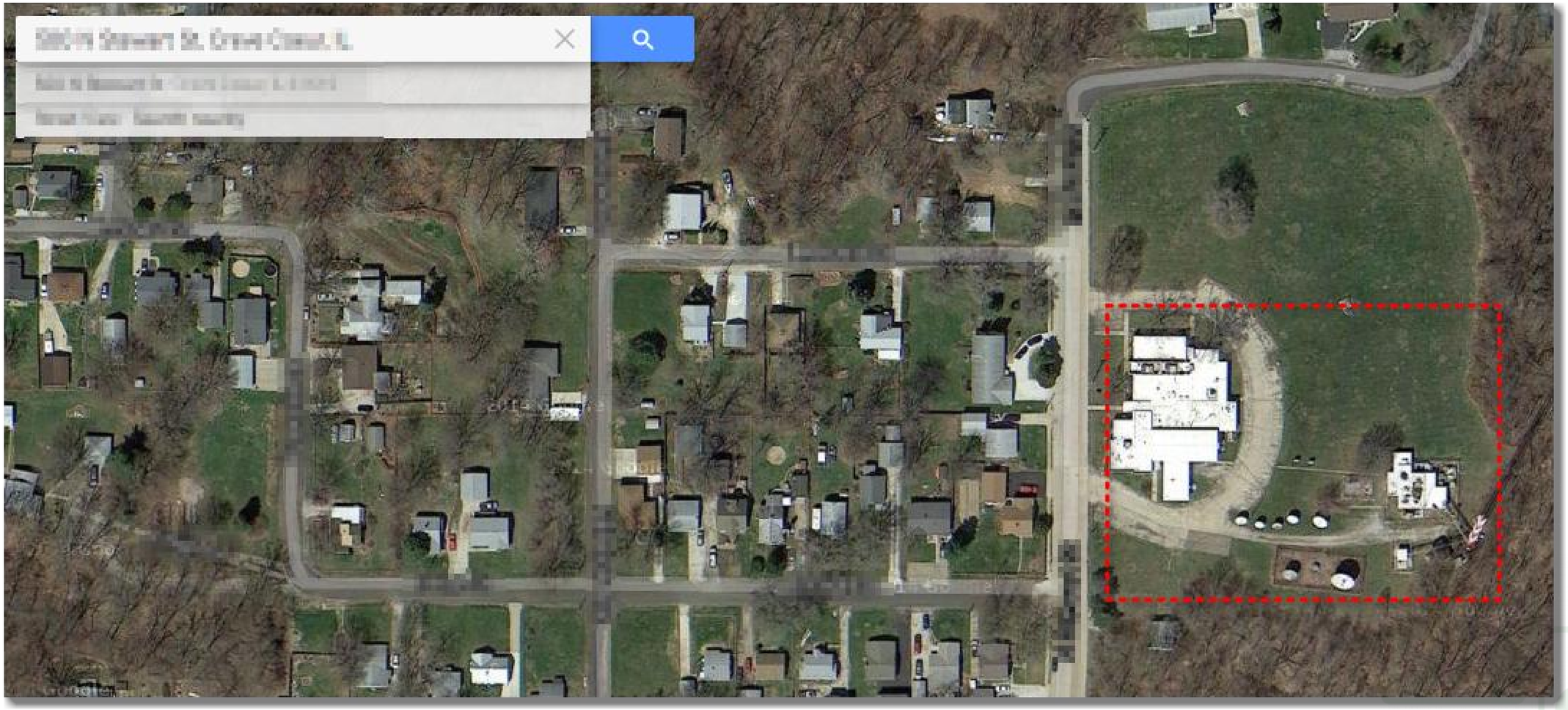
```
# Nmap 6.47 scan
Nmap scan report for development-device
Host is up, received user-set

PORT     STATE          SERVICE      REASON       VERSION
22/tcp   open           ssh          syn-ack      Dropbear sshd 0.51 (protocol 2.0)
23/tcp   open           telnet?      syn-ack
80/tcp   open           http         syn-ack      lighttpd 1.4.26
8080/tcp open           http         syn-ack      thttpd 2.25b 29dec2003
```

Find Devices



Description	Current Value	Set Value
Monitoring Method	Ethernet Eagle	--- Select ---
Primary IP Addr		192.168.[illegible]
Primary Port		5[illegible]0
Alternate IP Addr		17[illegible]3
Alternate Port		5[illegible]0
Listen Port		5[illegible]1
Automatic Update Interval	1 Hour(s)	--- Select ---

Locate Devices



Effects – Upgrade for free

Upgrade for free

July 24, 2013 – Oscilloscope
$800, 70MHz to $1600, 200MHz
Upgrade with a «Key-Gen»

```
/*
** XXXXX XXXXXX keygen / cybernet & the-eevblog-users
**
** to compile this you need MIRACL from https://github.com/CertiVox/MIRACL
** download the master.zip into a new folder and run
** 'unzip -j -aa -L master.zip'
** then run 'bash linux' to build the miracle.a library
**
** BUILD WITH:
**
** gcc rikey.c -I../MIRACL ../MIRACL/miracl.a -o rikey
**
…….....
```

Mitigation - Hardware

Hardware

- Use implemented security features
  - Fuses
  - Flash read protection
  - Tamper-switches
  - ...

Mitigation - Software

- Know and control your Software-Stack
  - Drivers / BSP
- Apply Software-Version-Control
- Release-Management
- Production Software QA
  - Clean builds (Leftovers)
  - Strip debug symbols
  - Remove development backdoors
  - Remove unneeded software
  - Harden / tighten

How about Qt?

Qt helps:

- As a 'vendor'
- As a community
- As a role model

Qt as a vendor

Qt is a mature and well monitored code base.

Many mistakes were already made and corrected.

Good integration avoids security problems caused by interfaces.

Collective maintenance of sensitive code is safer.

Track record of responses to alerts very good.

Qt as a community

Qt's source code is public, problems are found early and communicated openly.

Community members with security interest and knowledge keep an eye on things.

Nothing can be swept under the rug.

Help is available.

Qt as a role model

Institutionalized review process.

Collective code ownership.

Culture of security consciousness.

Systematic encapsulation and clean architecture.

Strict release process with audit trail.

Easily reachable, responsive, responsible and professional security response team.

Best practices

Don't bypass Qt, use safe infrastructure (strings, sockets, SSL, XML parsing, temporary files, byte arrays, database access, etc.).

Accept that any connected device can and will be attacked.

Systematic threat modelling
- => secure by default architecture
- => security as integral part of the development process
- => security analysis of finished product
- => security analysis of operational context

Don't trust vendors, no matter how big.

Mitigation - Other

- Less is more!
  - Is full IP-Stack really needed?
  - Remove debug-interfaces physically
  - Reduce used software libraries / products
- Do security reviews
  - Before someone else does
  - Bug-Bounty-Programms?
  - Incident handling?
- Take customers (data-) privacy seriously

Questions?