Network & Memory Bandwidth Regulation in a Soft Real-Time Healthcare Application*

M.D. Grammatikakis, G. Tsamis
P. Petrakis, A. Mouzakitis, M. Coppola
TEI of Crete
(Joint Work with VOSYS, STM)

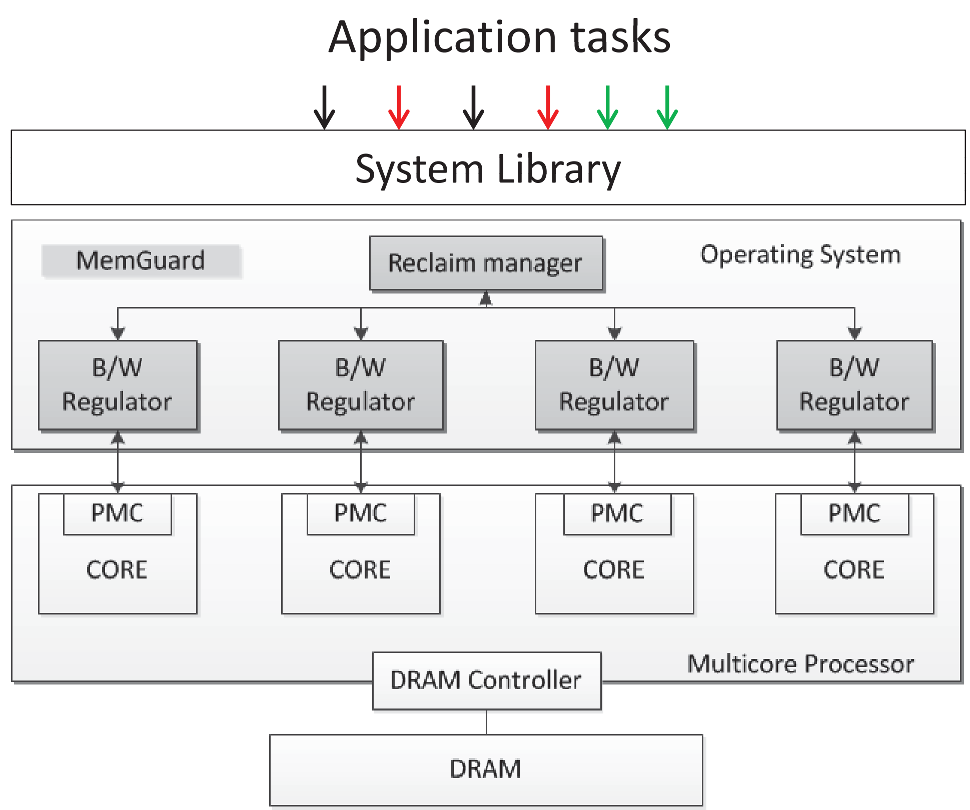
(*) Work supported by the EC through ICT Project DREAMS

OSPERT2017, Dubrovnik, Croatia

Introduction

- Holistic approach towards system resource management
- CPU, Memory and network bandwidth management can improve computation-, communication-intensive and/or memory-bound applications
  - ⇒ allocate memory resources in a fair manner
  - ⇒ avoid local saturation or monopoly phenomena
  - ⇒ avoid filling network capacity
  - ⇒ efficiently utilize available budget
- Concentrating on OS support, without additional hardware
- Not suitable for critical hard real-time operating systems

Related Work: Memory Management Policies



- MemGuard performs memory bandwidth regulation at core-level using performance counters monitoring the number of last-level cache misses

Genuine MemGuard: Principles & Extensions

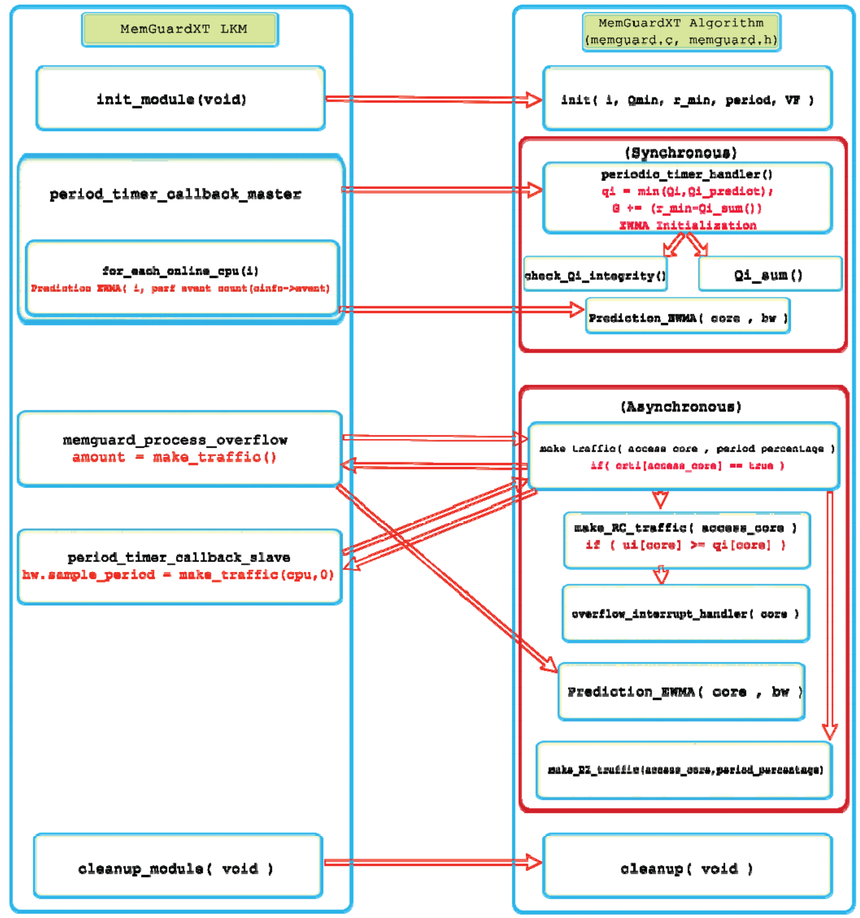
- Memguard allows sharing guaranteed bandwidth over several cores using a dynamic reclaim mechanism
  - cores are allocated at the beginning of each period part (or all) of their assigned bandwidth (history-based prediction)
  - cores donate the rest of their initially assigned bandwidth
  - global repository (called $\mathbb{G}$) contains donations
  - during period, a core may obtain additional budget from $\mathbb{G}$

Genuine MemGuard vs MemGuardXt

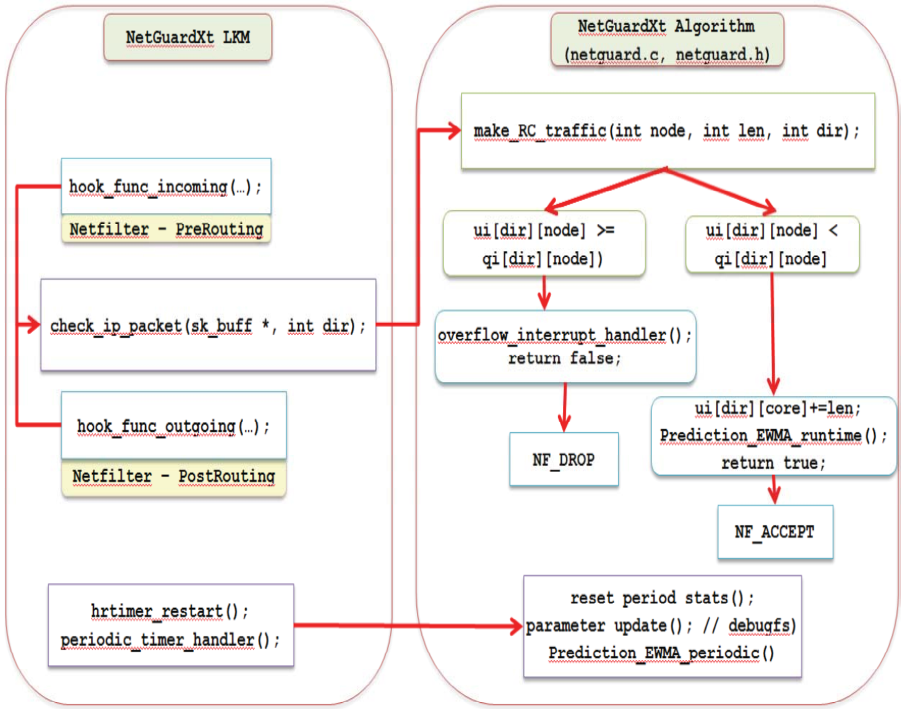
- A rate-constrained (RC) flow may steal guaranteed bandwidth from other RC flows
  - exhausting global repository and leading to guarantee bandwidth violations
  - potentially other RC flows have not yet demanded their full reservation
- Reservation-Only (RO) mode removes prediction and reclaiming, allocating RC traffic sources their full reservation in each regulation period
  - performs poorly, due to over-provisioning
    - a core cannot retrieve budget from ($r\_min - \Sigma Qi$), if $\Sigma Qi < r\_min$
- Limited adaptivity for predicting memory bandwidth requirements
- MemGuardXt algorithm
  - supports modularity (multiple LKM instances)
  - provides a hard guarantee option called Violation Free or **`VF`** by restricting reclaiming from G by one or more rate-constrained cores, if, as a result, it can lead to a guarantee violation for another RC-flow
  - improved prediction of future bandwidth requirements using EWMA

MemGuardXt LKM

- `init_module` & `cleanup_module` do initialization & memory cleanup
- `Prediction_EWMA` updates the bandwidth consumed by each core
- `periodic_timer_handler` resets statistics and reassigns estimated bandwidth per core
- `make_traffic`
  - called at start of period to update bandwidth consumed in previous period
  - called on the fly (by asynchronous calls) when more bandwidth is required



NetGuardXt LKM



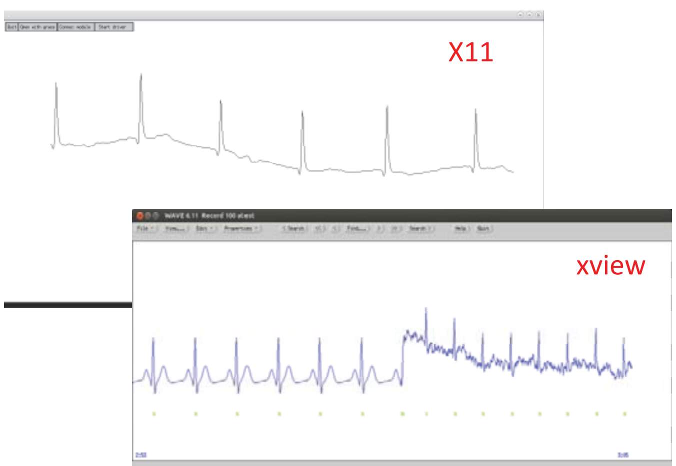
- NetGuardXt uses custom netfilter hooks to drop, accept or buffer packets
- Similar regulation algorithm
- Incoming & outgoing flows controlled by the same LKM
  - parameters configured on the fly, independently for each flow direction
  - similar API to MemGuardXt
- Each incoming/outgoing packet checked using `make_rc_traffic`

Hospital Media Gateway (HMG)

- MemGuardXt/NetGuardXt LKMs evaluated in correlation to real-time using an actual mixed-criticality use case with
  - critical medical tasks associated with soft real-time ECG analysis
  - non-critical video streaming for delivering premium content to patient
    - evolving traditional linear system (TV $\Rightarrow$ infotainment/smart devices)
    - quality-of-delivery via NetGuardXt

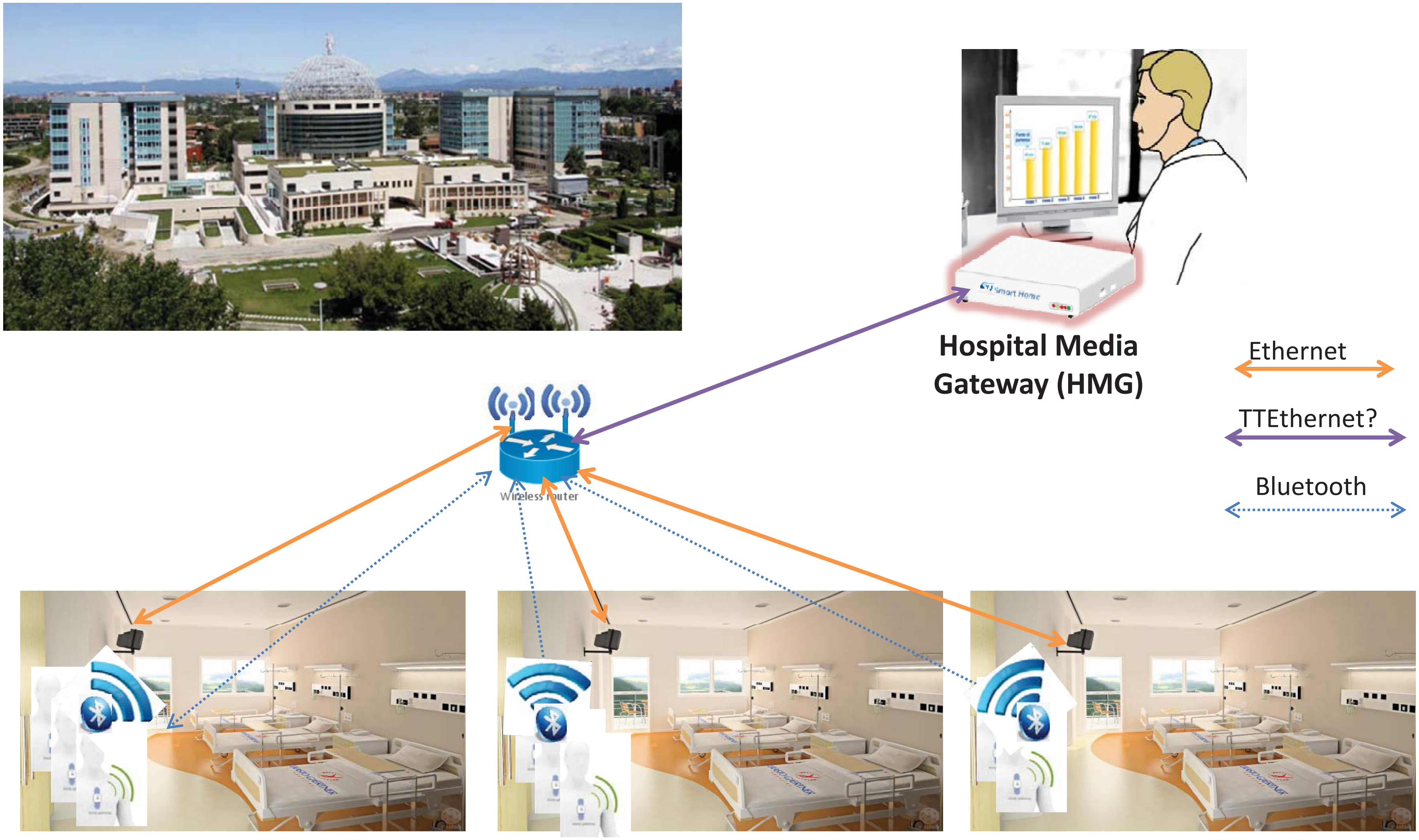
```
root@linaro-ubuntu-desktop:~/netguard_driver# echo "1500 50 300 1200" > /sys/kernel/debug/netguard/netguard_config
root@linaro-ubuntu-desktop:~/netguard_driver# echo "1500 50 1200 300" > /sys/kernel/debug/netguard/netguard_config
root@linaro-ubuntu-desktop:~/netguard_driver# 
```

Real-Time Healthcare Application Extension

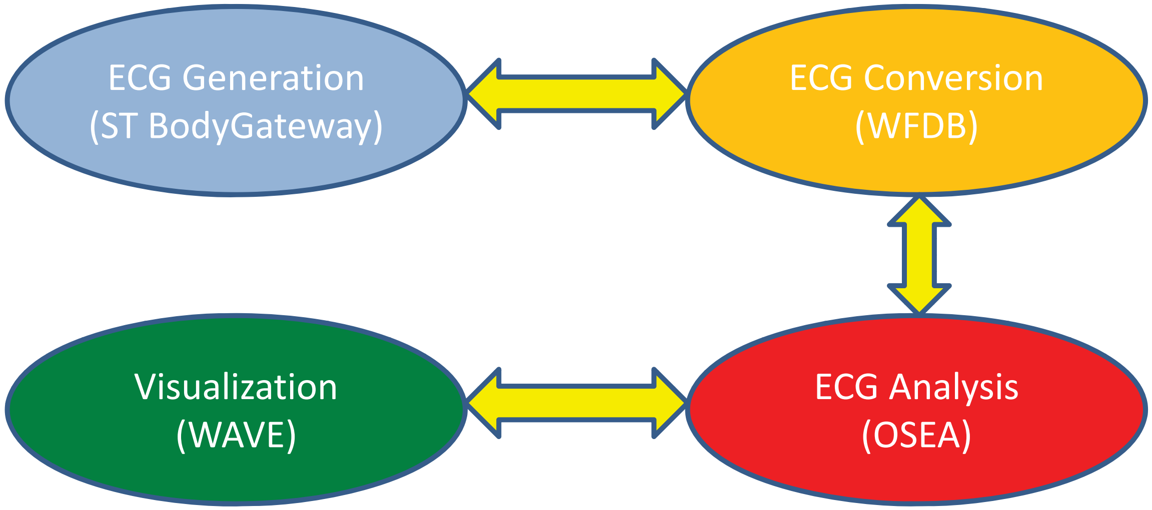


- ST BodyGateway (BGW)
  - body worn: ECG, respiration rate, accelerometer, holter …
  - BT 3.0 connection
  - GNU/Linux driver developed
  - port to ARM v7 (in future ARM v8)
- Open source ECG analysis (soft RT) to detect arrhythmias
  - filters detect & classify beat: N, V, …
  - annotated ECG (Xview/X11-based)

Target Application: In-Hospital Scenario

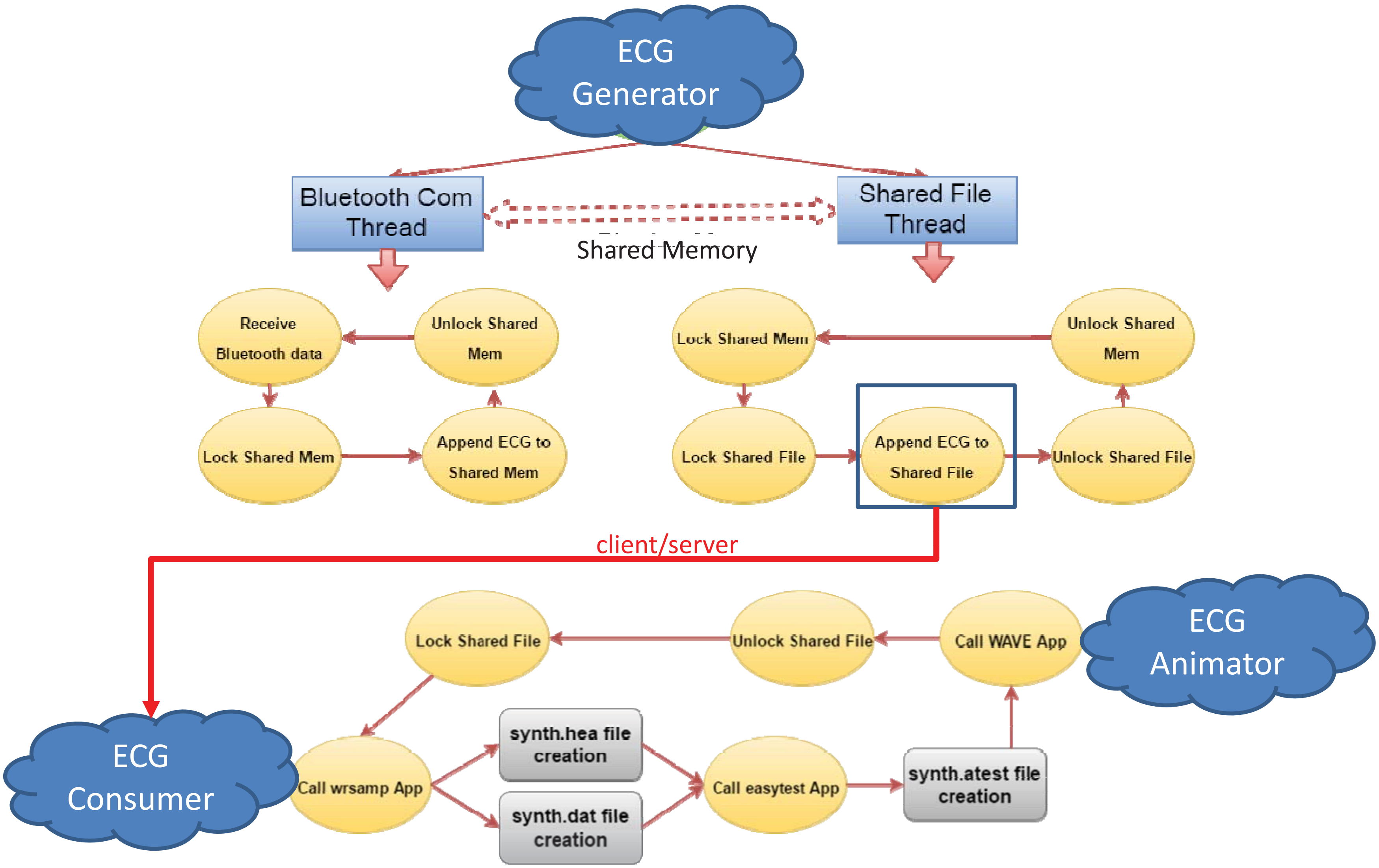


Open Source ECG: Online ECG Analysis (RT)

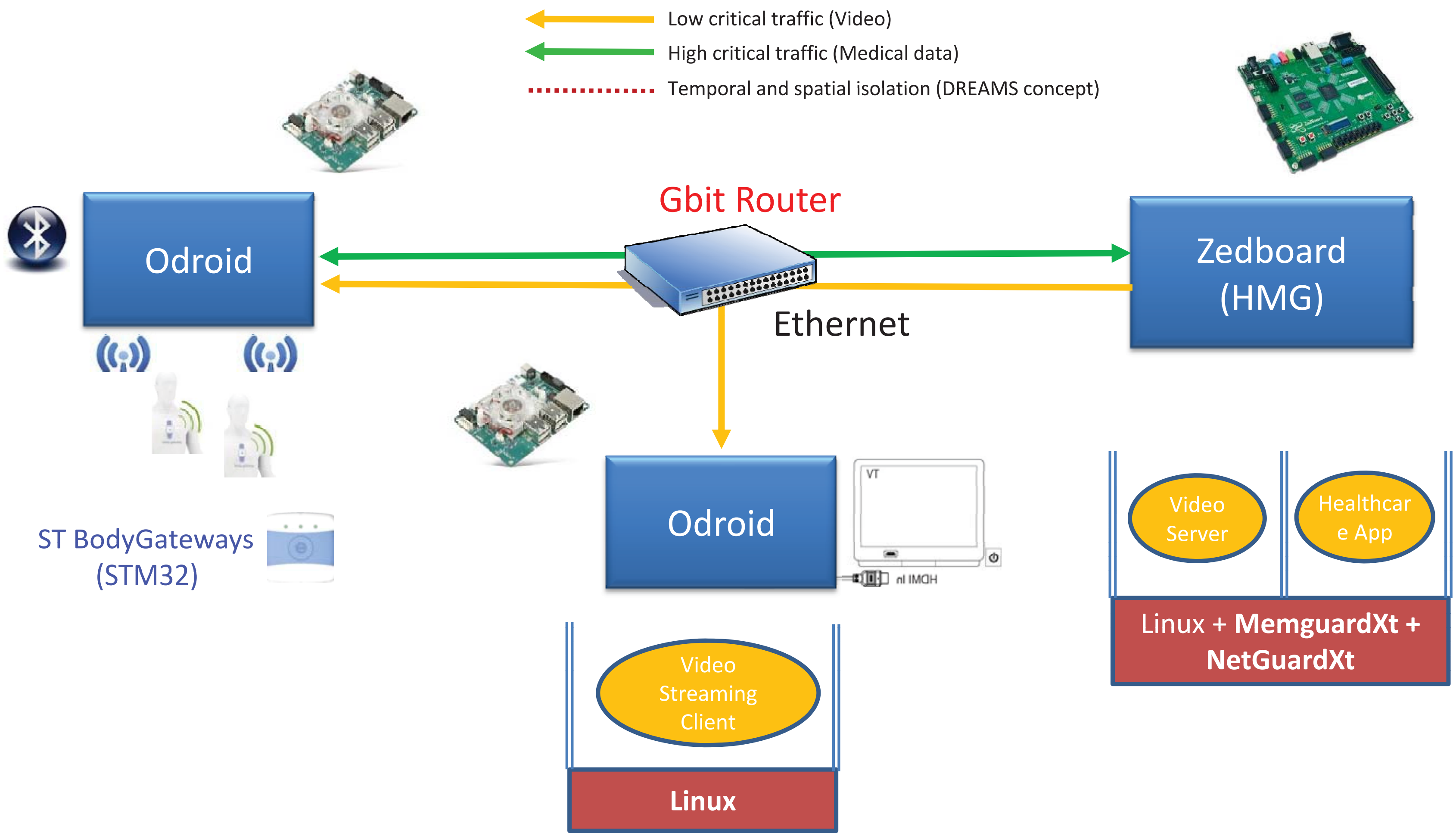


- Concurrent ops (file locking, sem/mutexes)
  - ECG Conversion to std format
  - ECG Analysis via filters to detect/classify beats - EC13-compliant
    - QRS positive predictivity ~99.8% for normal (N) & ventricular beats (V)
    - 3min training (we use ECG synthesis, real ECG depends on age/sex)
  - Visualization of heartbeat with (N, V) annotation

Distributed Embedded System



Experimental Framework



Hardware Configuration & Mapping

- ECG traffic arrives to HMG from Odroid (two BGWs) via Ethernet
- Video-on-demand traffic arrives to HMG via Ethernet and is distributed to clients via streaming

- HMG (Zedboard) has two ARMv7 Cortex-A9 cores
  - ECG analysis mapped to CPU0 (server rx, consumer, animator)
  - Video-on-demand service runs on CPU1

- NetguardXt regulates only incoming ECG/Video traffic
  - Outgoing flows not considered , since memory bandwidth increases very slightly (1-2 MB/s) with video streaming

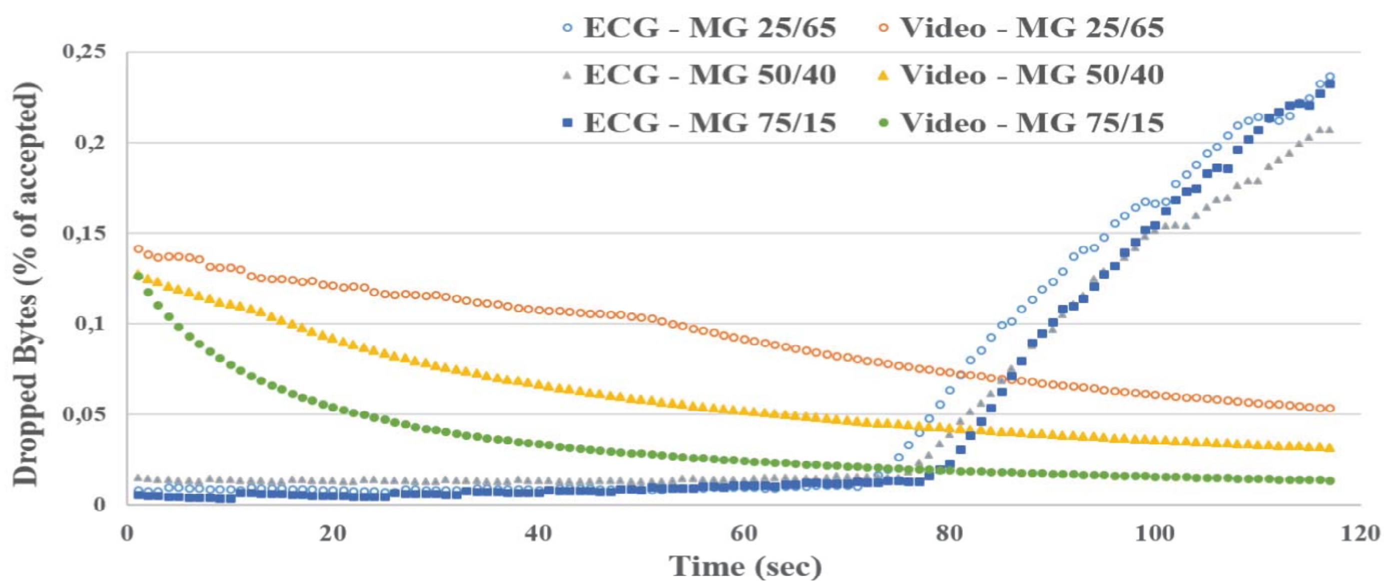
LKMs: Configurations and Runtime Scripts

- Initial **`MemGuardXt`** configuration (static)
  - **`period=1ms, i=2, λ=0.2, r_min=Q0+Q1=90MB/s, Q_min=50MB/s`**
- Initial **`NetGuardXt configuration`** (static)
  - **`period=1s, i=2, λ=0.2, r_min=Q0+Q1=70KB/s, Q_min=1MB/s`**

  `Q0+Q1` based on initial ECG analysis/video experiments in isolation

- Simultaneous memory/network bandwidth regulation using one of 3 scripts
  - **`MG 25/65`**, meaning **`MemGuardXt Q0/Q1 = 25/65`** *(i.e. in favor of Video)*
  - **`MG 50/40`** (i.e. a more balanced ratio)
  - **`MG 75/15`**, meaning **`MemGuardXt Q0/Q1 = 75/15`** *(i.e. in favor of ECG)*
- Each such script runs **`2m`** and *periodically, every **`20s`**, reconfigures **`NetGuardXt Q0/Q1`*** using the sequence **`{18/72, 16/74, 14/76, 12/78, 10/80, 8/82}`**
  - this gradually decreases assigned network budget for ECG (in favor of Video)
- **`VF`** mode used by default for both guards
  - exception last figure, where **`MemGuardXt`** is used in N**`on-VF`** mode

Experimental Results: Kernel Logs (NetGuardXt)



- Dropping ECG network bandwidth via NetGuardXt from 18kB/sec to 8KB/sec (in 20 sec intervals), increases cummulative ECG drop rate and decreases the drop rate of video traffic

Experimental Results: ECG Analysis

- Running **MG 75/15 script**
- in **VF** or **Non-VF**
- **Total execution time**

⇒ **~30% for receiving ECG** (server rx)

~20% for data conversion (wrsamp)

~50% for ECG analysis (easytest)

- Small variations due to file locks

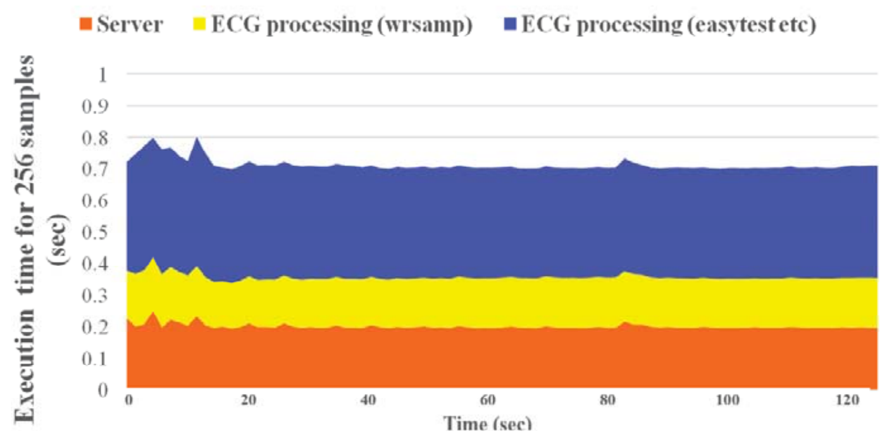
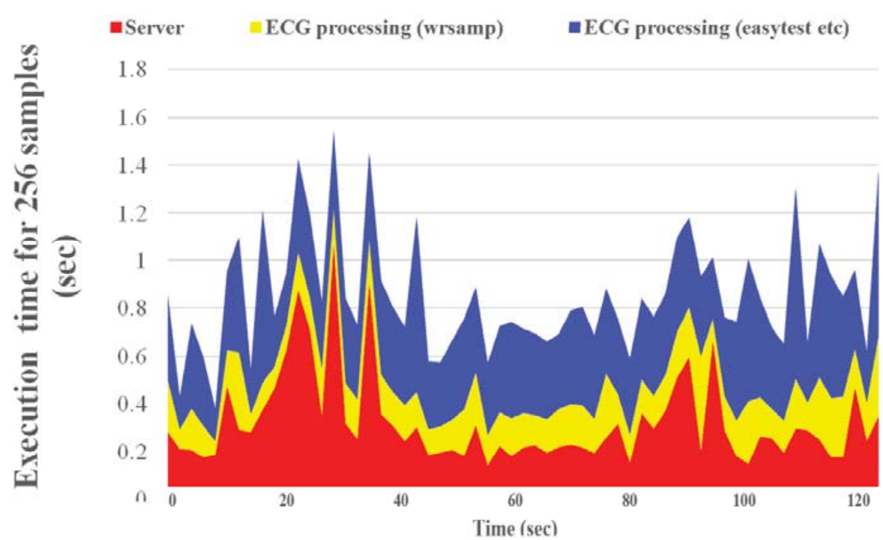
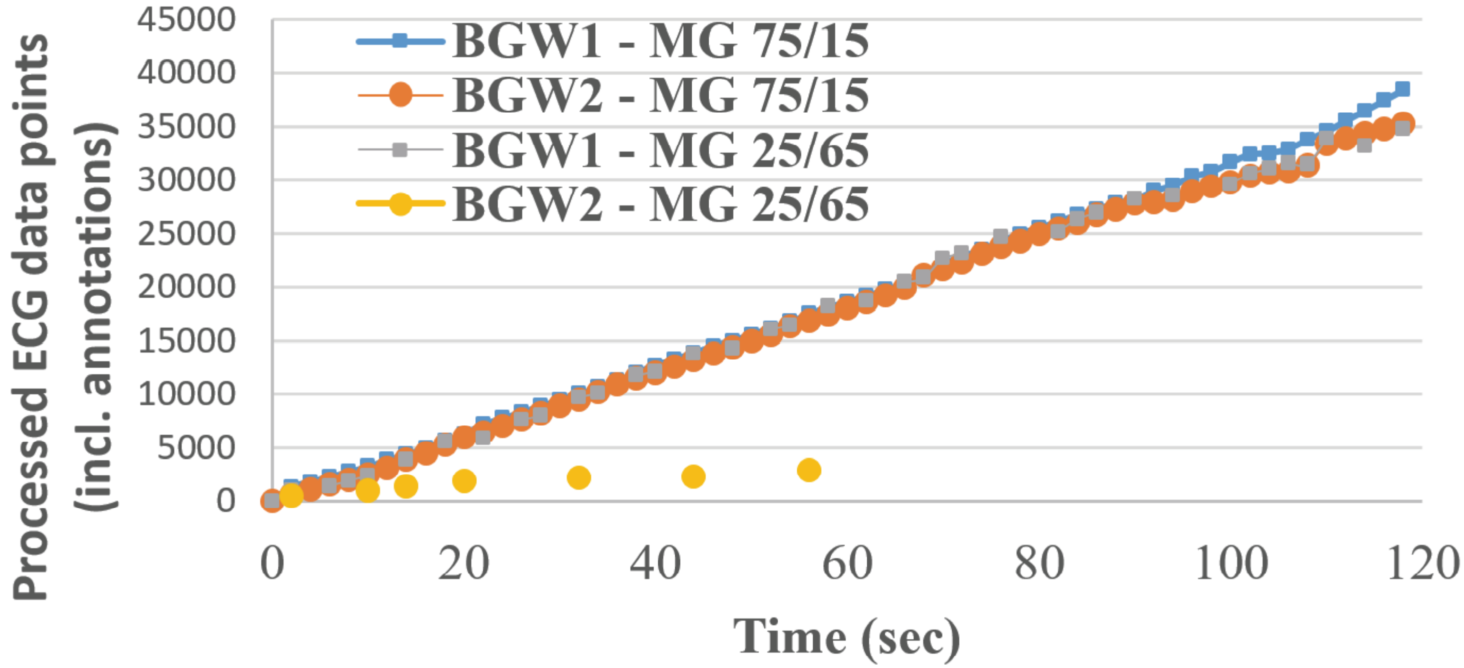


Fig. 10. Delays at Home Media Gateway for ***MG75/15*** *script*, ***VF*** *mode*



- With **`MemGuardXt Non-VF`** mode, HMG cannot cope w soft real-time (~75K guarantee violations)

Experimental Results: Scalability



- For **`MG 25/65`**, BGW2 devices has completely stopped due to low memory bandwidth; in other experiments, both BGWs lag
- For **`MG 75/15`**, both BGW devices operate in soft real time.

Summary & Future Work

- Extend MemGuard's memory bandwidth regulation policies with
  - adaptivity through EWMA
  - violation free operating mode
  - highly modular approach
    - extension to network bandwidth regulation module (NetGuardXt)
  - same "memguard" prototype used in multiple module instances
- Mixed-criticality use case on a hospital media gateway prototype
  - *soft real-time ECG analysis*
  - *video-on-demand streaming*
- Control of network/memory bandwidth can improve ECG processing
- Future Work
  - use ARMv9 Juno board & time-triggered TTEthernet switch (rtwifi?)
  - MemGuard implementation at the level of Linux scheduler

Main References

- H. Yun, G. Yao, R. Pellizzoni, M. Caccamo, and L. Sha, “Memguard: Memory bandwidth reservation system for efficient performance isolation in multi-core platforms,” *in Proc. IEEE Symp. Real-Time and Embedded Tech. and Appl.*, 2013, pp. 55—64.
- H. Yun, G. Yao, R. Pellizzoni, M. Caccamo, and L. Sha, “Memory bandwidth management for efficient performance isolation in multicore platforms”, *IEEE Trans. on Computers*, **65 (2)**, 2016 pp. 562—576.
- B. Akesson and K. Goossens, “Architectures and modeling of predictable memory controllers for improved system integration,” *in Proc. Design, Automation Test in Europe Conf.*, 2011, pp. 1–6.
- G. Tsamis, S. Kavvadias, A. Papagrigoriou, M.D. Grammatikakis, and K. Papadimitriou, “Efficient bandwidth regulation at memory controller for mixed criticality applications”, *in Proc. Reconfigurable SoC*, 2016, pp. 1—8.
- Soft Real Time ECG Analysis and Visualization

  https://physionet.org/works/SoftRealTimeECGAnalysisandVisualization