SWN
Softwarized & Wireless Networks Research Group

mec

MONROE

GSRT
GENERAL SECRETARIAT FOR RESEARCH AND TECHNOLOGY

HELLENIC REPUBLIC
UNIVERSITY OF MACEDONIA

60 YEARS 1957-2017

Elastic Content Distribution based on Unikernels and Change-Point Analysis

Lefteris Mamatas, Assistant Professor

emamatas@uom.edu.gr

University of Macedonia, Greece

ETIS-ICI, Paris, France
16/3/2018

http://swn.uom.gr

The SWN Research Group

The Softwarized & Wireless Networks (SWN) research group (http://swn.uom.gr) was established in 2016 as a part of the Computer & Network Systems Technologies (CNST) laboratory of the Department of Applied Informatics, University of Macedonia, Greece (http://www.uom.gr).

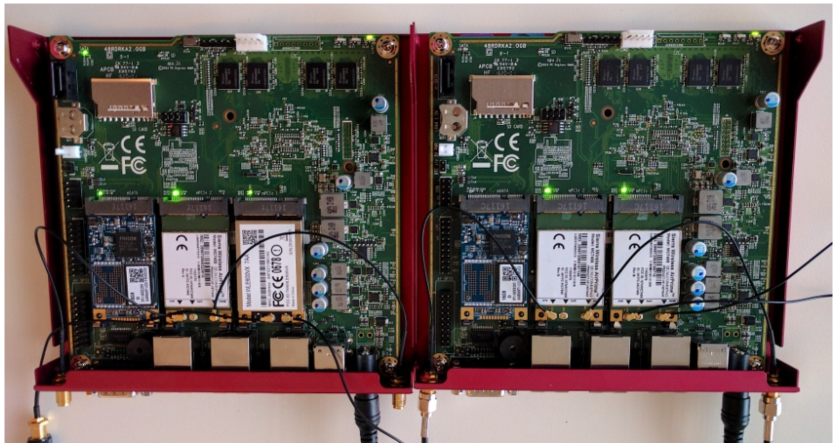
Research Areas: Software-Defined Networks, Network Function Virtualization, Cloud Networking technologies and their integration with the network edge, e.g., Mobile Edge / Fog Computing.

Active Research Projects: NECOS H2020 (Lightweight Clouds & Cloud Slicing), MONROE H2020 OC2 (Lightweight Clouds & Mobility)

Research group: one + three academics, one post-doc, four PhD students and one software engineer.

The MONROE Project & Platform

- EaaS platform for measurements with MBB networks
  - EU H2020 project www.monroe-project.eu
- 200 multi-homed Linux nodes across Europe
- Each connected to 3 commercial cellular networks
- Open to external experimenters
- A very rich database with measurements



Outline

- Objectives of this work
- Why Unikernels?
- The MEC Platform
- Use-Cases
- Platform Details and Results
    - Modular Platform Orchestration
    - DNS-based Load Balancing
    - Change-Point Detection
    - Multi-homing Capability
    - Synthetic IoT Mobility
- Related Works
- Conclusions

In the context of the Multihoming with Ephemeral Clouds on the Move (MEC) MONROE H2020 (grant agreement number 644399) OC2 Project.

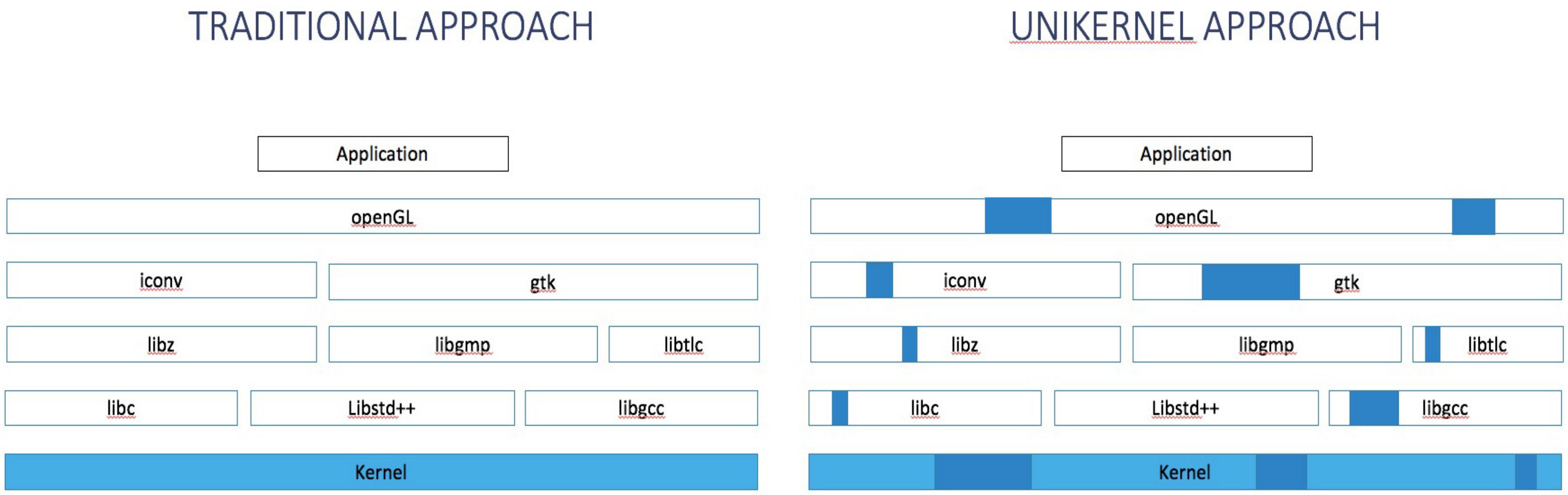
MEC Platform Objectives

- Goal of MEC platform: ***"To bring efficiently together Mobile Networks with Lightweight Clouds, in a Mobile Edge Computing context".***

- We experiment with:
  - Intelligent Orchestrated Lightweight Cloud Resources
  - Novel detection and forecasting mechanisms for the dynamic network conditions
  - Multi-homing Capabilities

- Enabling aspects:
  - The MONROE experimentation facilities
  - Unikernels
  - Change-point detection

Why Unikernels?



- **Unikernels** are extremely fast and lightweight VMs:
  - specialized at compile-time into standalone kernels,
  - acting as individual SW components,
  - sealed against modification when deployed in a cloud platform,
- They can even boot up with a DNS request or TCP SYN packet.

Unikernels vs Containers

The ***Unikernels*** are single-purpose appliances that are:

- specialized at compile-time into standalone kernels;
- acting as individual software components; and
- sealed against modification.

Containers wrap-up application software in a complete filesystem that contains:

- code, runtime, system tools and system libraries

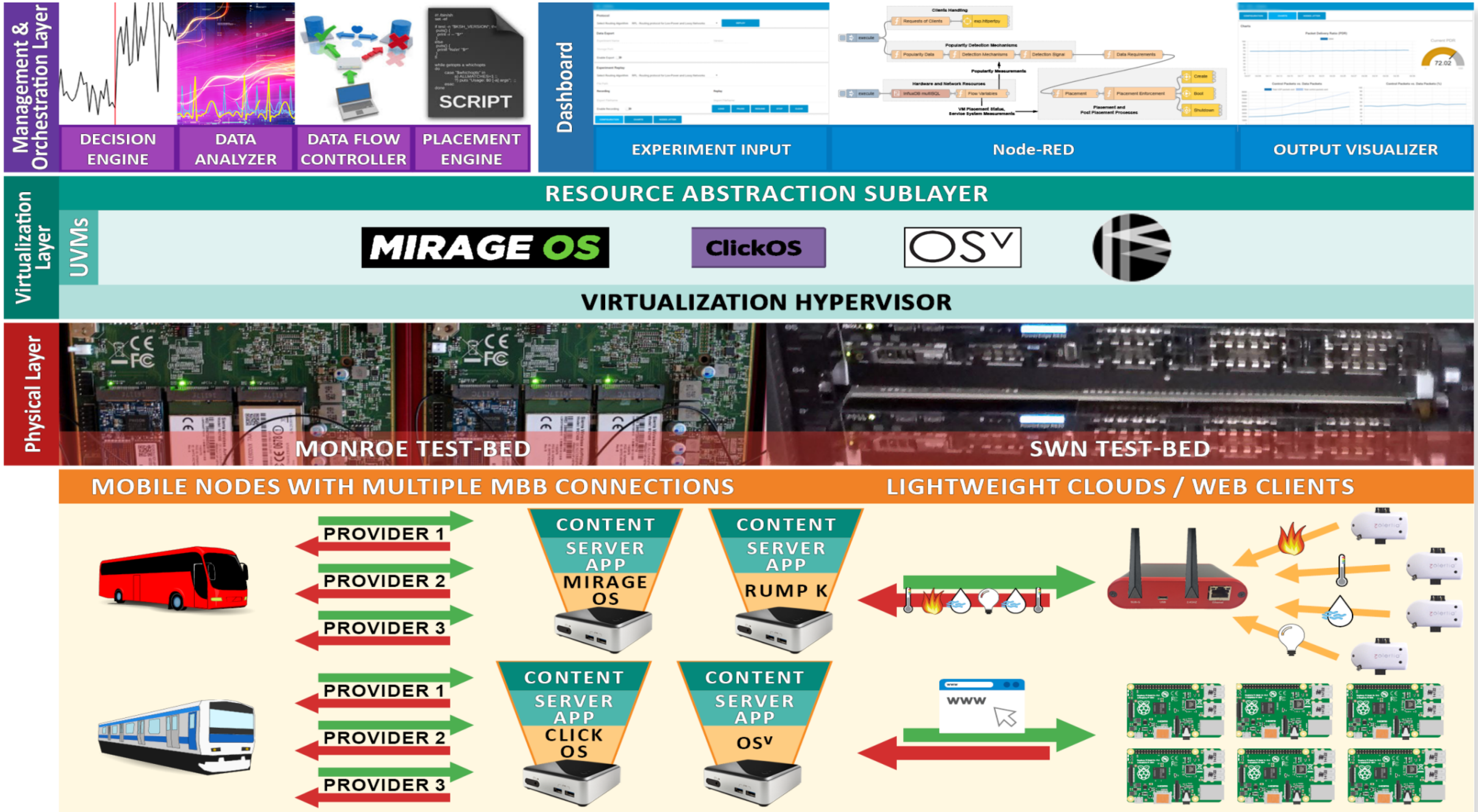
Unikernels	Containers
Isolate down to the hypervisor	**Share host kernel**
Specific component	**Minimal operation systems**
boot up in few ms	**Boot up of hundred of ms**

Indicative Statistics

Technology Type	Time (msecs)	Size (MBs)	Usage (MBs)
standardvm.xen	6500	913	112
standardvm.kvm	2988	913	82
Container	1711	61	3.8
tinyx.kvm	1081	3.5	30
tinyx.xen	431	3.7	31
unikernel.osv.kvm	330	12	52
unikernels.minios.xen	31	2	8

Source: https://tools.ietf.org/html/draft-natarajan-nfvrg-containers-for-nfv-03

The MEC Platform Architecture



Investigated Scenarios: 1) Elastic CDNs

- Conditions:
  - Background traffic requesting content
  - Mobile web clients requesting popular content delay because of:
    - MBB network inefficiencies
    - overloaded web servers

- Tackle from the **Cloud viewpoint**:
  - Content-popularity change detection
  - Efficient placement / replication of content VMs
  - Assigning mobile clients to better VMs

- Tackle from the **MBB network viewpoint**:
  - Choosing the most appropriate provider
    - Using change point detection & stochastic models

Investigated Scenarios: 2) Fog Computing

- Clients request measurements from IoT nodes
  - e.g., temperature, humidity, pressure, etc.

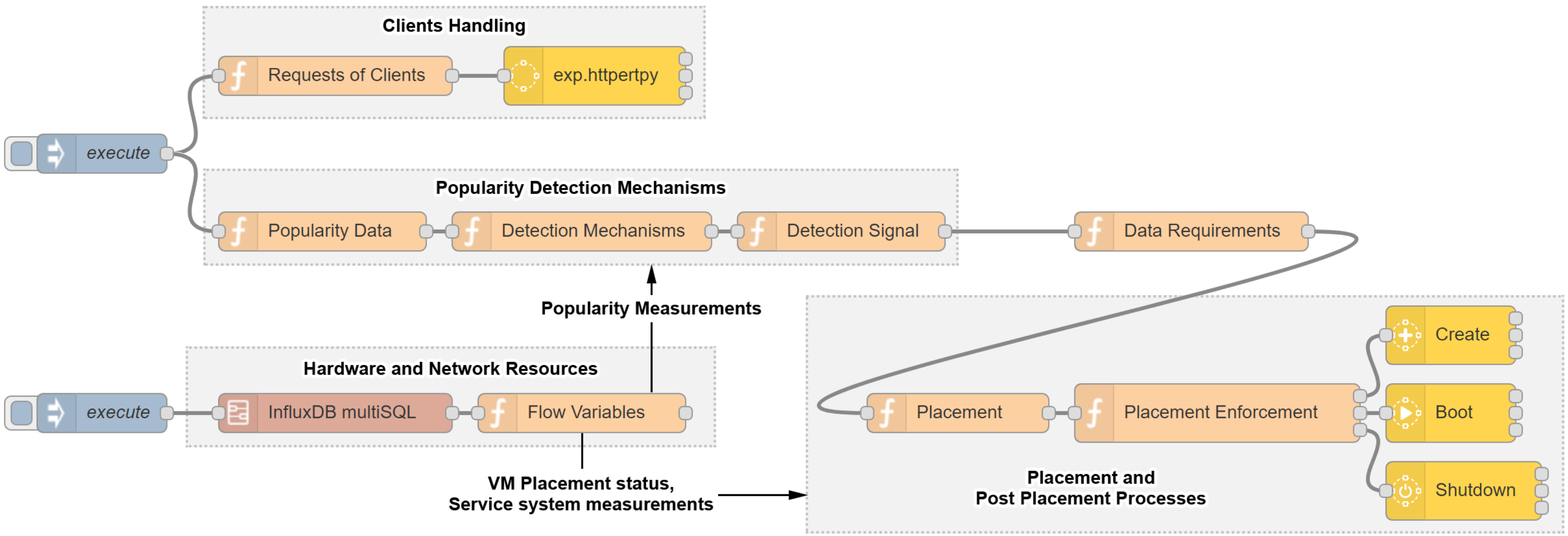
- IoT nodes or clients may be mobile

- Lightweight VMs are being deployed to assist the communication
  - e.g., data collectors, lightweight DBs for measurements etc.

- Improve communication over the **MBB & fixed network**
  - e.g., multi-homing capabilities, lightweight VM caching

- Improve communication over the **IoT network**
  - SDN-inspired IoT protocols, e.g., mobility aware

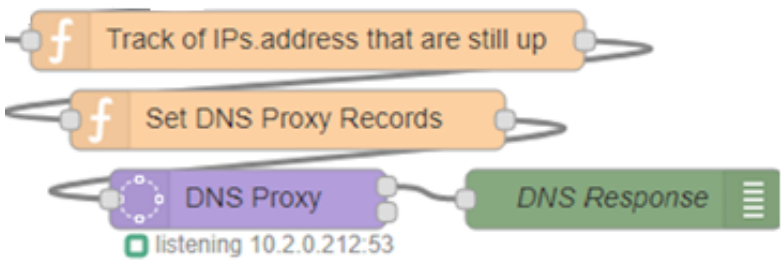
Modular Platform Orchestration

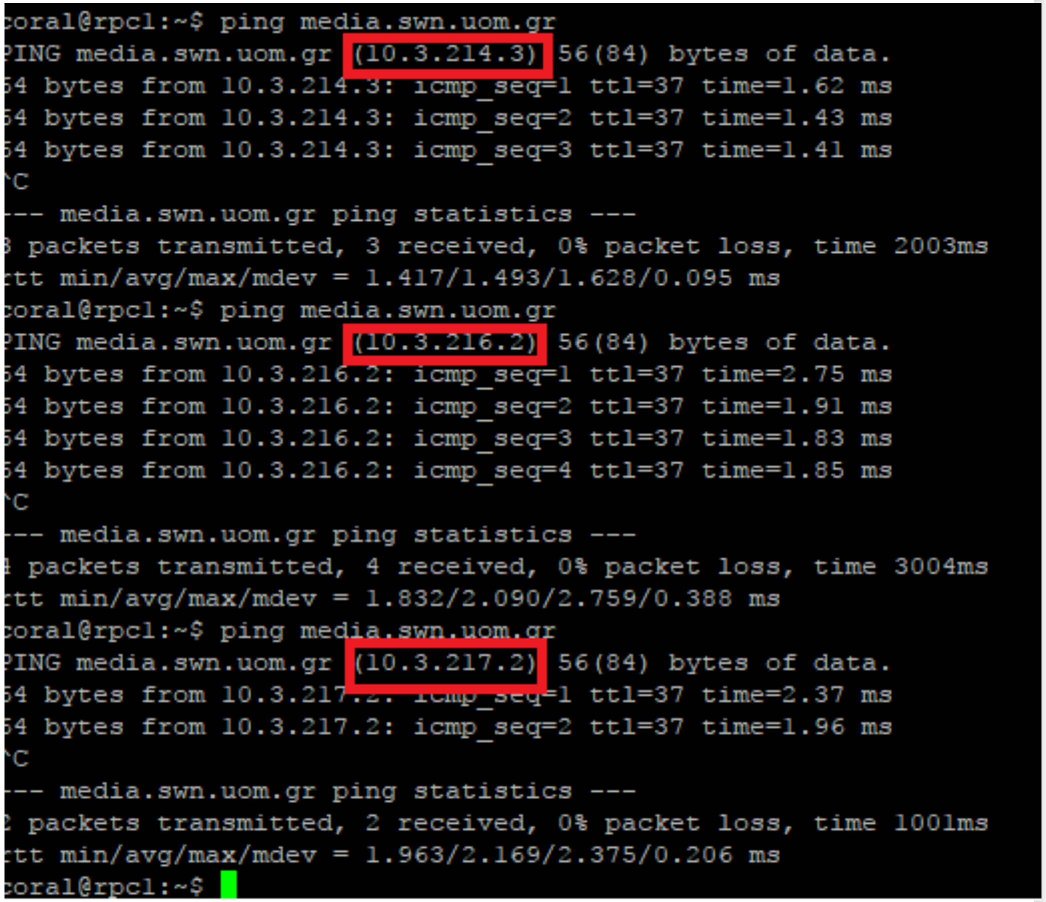


- Platform orchestration & management features implemented as Node-RED nodes:
  - Unikernel VM placement decision algorithms and VM control features
  - DNS-based load balancing
  - Content popularity change detection

- We cache content in multiple unikernels deployed on the fly, when **content becomes popular** or to **improve mobile nodes performance**

DNS-Based Load Balancing





Change-Point Detection Analysis

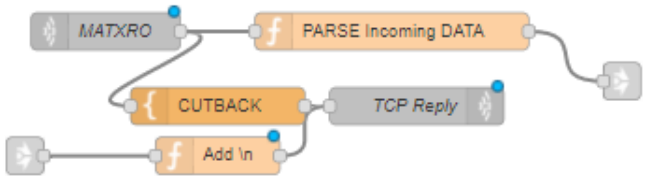
- Classification of change-point problems:
  - How **data** is received
    - Off-line data collection, i.e., retrospective
    - On-line data collection, i.e., sequential
  - **Statistical information** of the r.v. sequence
    - Non-parametric or parametric methods
  - **Type** of change-point:
    - Mean or 2nd order characteristics

- Fundamental aspects
  - Change-point detection problems are phrased as hypothesis tests
  - Test statistics may be viewed as two-sample tests adjusted for the unknown break location, leading to max-type procedures
  - Asymptotic behavior is derived to obtain the tests' critical values
  - For the case of multiple changes, a hierarchical algorithm is involved

Our Approach to Change-Point Detection

- We perform **change-point detection** for:
  - detection of **content popularity changes**, i.e., place more content VMs
  - detection of **changes in the MBB performance**, i.e., switch to a different provider

- **Main requirements:**
  - **Low complexity to match the main characteristics of Unikernels**
    - resource-efficiency
    - quick estimation, e.g., low convergence time
    - non-parametric framework
  - **Be able to estimate**:
    - the existence, direction and rough magnitude of change
    - the number of changes
  - Operate in an **on-line manner**
  - Work with **real youtube content popularity measurements**



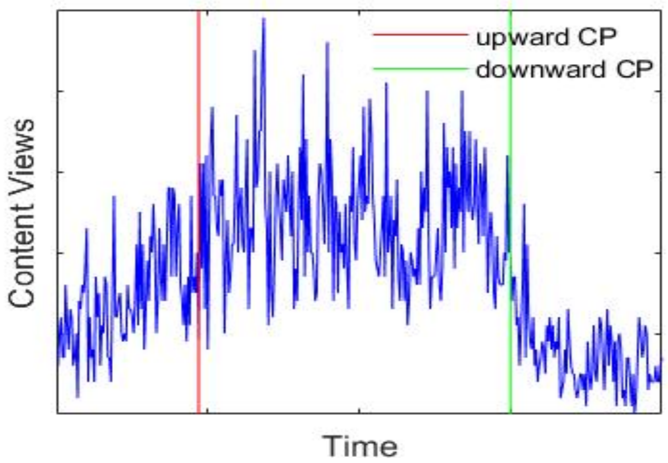
Our Approach to Change-Point Detection

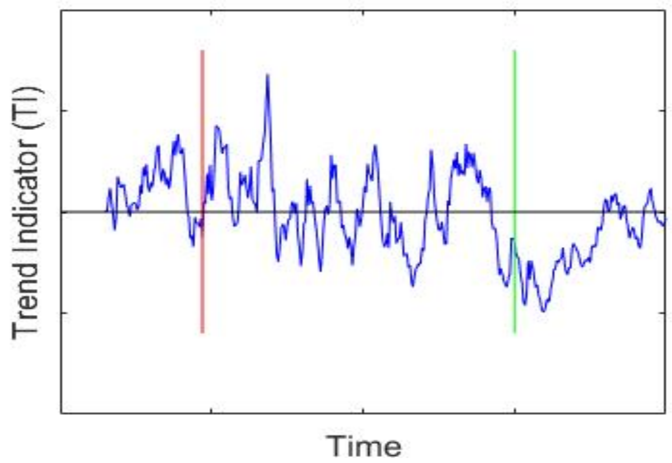
- **Detect the existence of change** based on:
  - a retrospective CUSUM type detector for a null hypothesis testing of no change, and
  - the long-run covariance estimator Newey-west to capture serial dependence and fluctuation.

- **Estimate the number of changes** using:
  - our own highly flexible and computational efficient heuristic segmentation algorithm,
    - combines the BS and ICCS algorithms, to tackle the overestimation issue of BS, with the CUSUM test statistic

Our Approach to Change-Point Detection

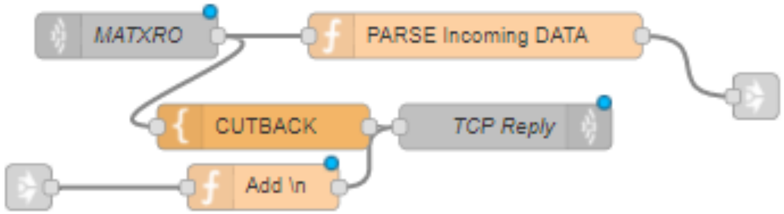
- **Detects the direction of changes** using:
  - the Moving Average Convergence Divergence (MACD) indicator
    - subtracts a shorter exponential moving average (EMA) filter from a longer one, i.e., the MACD series
    - The difference between the MACD and an even shorter EMA of the MACD, reveals the direction





Our Approach to Change-Point Detection

- **On-line change detection mechanism** based on:
  - a sequential change-point method based on the CUSUM detector,
    - i.e., identifies if the no-change assumption for period m, holds as new data k become available.
  - the procedure stops with a stopping rule based on a sensitivity parameter $\gamma \in [0, \frac{1}{2})$, with potential values:
    - 0 may lead to **a slower change detection**
    - 0.45 leads to a **quicker detection**, but more sensitive to **false alarms**
    - 0.25 is a **good compromise**



Offline Change-Point Detection

- Let $\{Y_t : t \epsilon \mathbb{Z}\}$ be a sequence of d- dimensional random vectors, where,

$$Y_j = \mu_t + \varepsilon_t$$

- Null hypothesis is

$$H_0 : \mu_1 = \cdots = \mu_n = \mu$$

against the hypothesis of at least one change.

Horvath et al. (1999)

CUSUM process Z_n is defined as,

$$Z_n(k) = \frac{1}{\sqrt{n}} \left(\sum_{t=1}^{k} Y_t - \frac{k}{n} \sum_{t=1}^{n} Y_t \right)$$

Offline Change-Point Detection

Test statistic is given by,

$$M_n = \max_{1 \leqslant k \leqslant n} Z_k' \widehat{\Omega}_n^{-1} Z_k.$$

Asymptotic behaviour under H_0,

$$M_n \Rightarrow \sup_{0 \leqslant t \leqslant 1} \sum_{l=1}^{d} B_l^2(t) \quad (n \to \infty)$$

Point estimator,

$$\widehat{cp} = \frac{1}{n} argmax_{1 \leqslant k \leqslant n} M_n$$

Long-run Covariance Estimation

$$\Omega = \sum_{t \epsilon \mathbb{Z}} Cov\left([Y_0 Y_0'], [Y_t Y_t']\right)$$

$$\widehat{\Omega}_n \rightarrow \Omega \quad (n \rightarrow \infty)$$

Two general approaches for the non parametric estimation of $\widehat{\Omega}_n$.

- bootstrap
- kernel based

Long-run Covariance Estimation

Non-parametric Newey-West estimator [Newey et al. (1989)],

$$\widehat{\Omega}_n = \widehat{\Sigma}_0 + \sum_{h=1}^{H} k\left(\frac{h}{H+1}\right)\left(\widehat{\Sigma}_h + \widehat{\Sigma}'_h\right)$$

- $k(.)$, the Bartlett kernel, where,

$$k(x) = \left\{ \begin{array}{ll} 1 - |x|, & \text{for } |x| \leq 1 \\ 0, & \text{otherwise} \end{array} \right\}$$

- Σ, the covariance matrix
- H, the bandwidth $H = \lfloor n^{1/3} \rfloor$ (minimize overestimation)

Segmentation Algorithms

Address the issue of multiple change points detection.

- Class of Binary segmentation algorithms:
  - first, search for a single change point in the whole sequence.
  - then the sequence is split in two subsequences until no more points are detected.
- Our approach combines:
  - ICSS algorithm [Inclan et al. (1994)].
  - BS algorithm [Vostrikova et al. (1981)].
  - the CUSUM test statistics previously presented.

Online Change-Point Detection

Assumption: $\mu_1 = \cdots = \mu_m$.

$$H_0 : \mu_{m+1} = \mu_{m+2} = \cdots$$

$$H_1 : \mu_{m+1} = \cdots = \mu_{m+k^*-1} \neq \mu_{m+k^*} = \mu_{m+k*+1} = \cdots$$

where $1 \leqslant k^* < \infty$ denotes the unknown time of change.
m: the length of the training period (model is assumed to be stable).

Online Change-Point Detection

Stefan Fremdt. (2014)

CUSUM detector is given by,

$$Q(m,k) = \sum_{i=m+1}^{m+k} Y_i - \frac{k}{m}\sum_{i=1}^{m} Y_i$$

Stopping time,

$$\tau_m = min\{k \geq 1 : Q(m,k) \geq \widehat{\omega}_m c_{1,a}\, g(m,k)\}$$

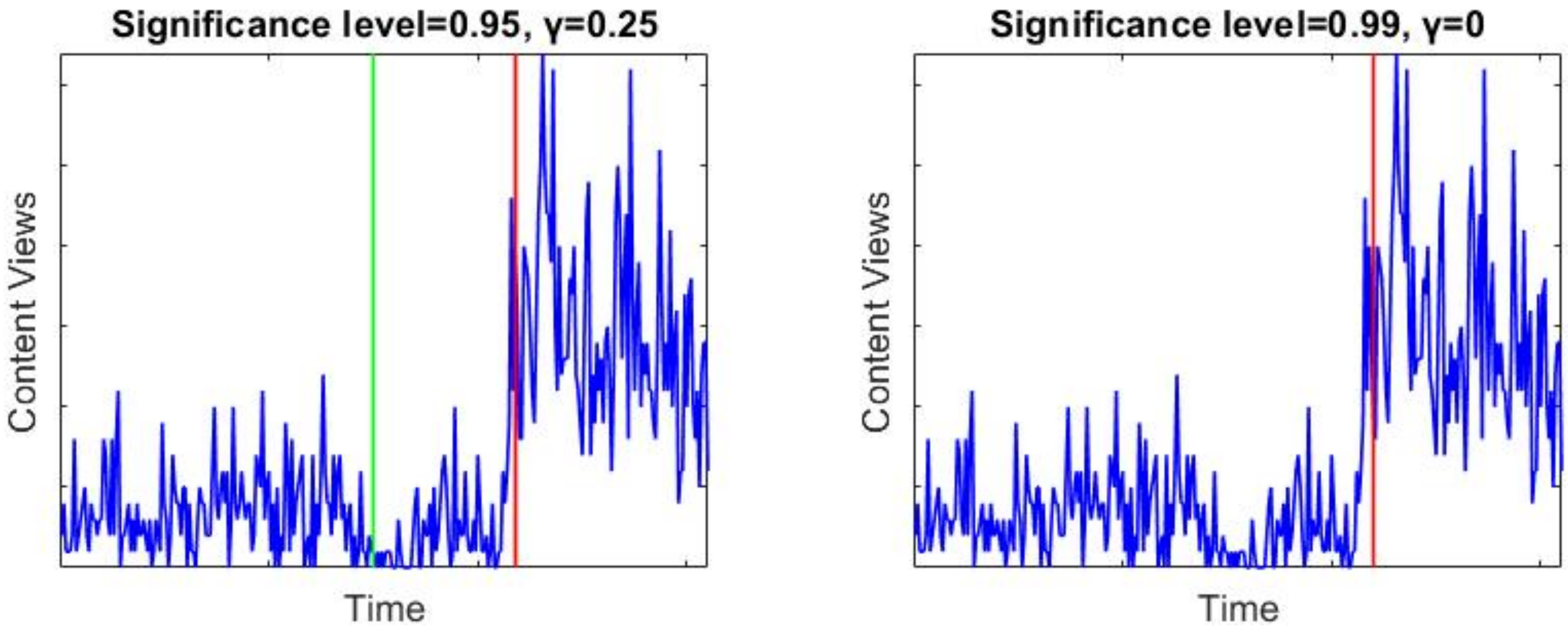
$$g(m,k) = \sqrt{m}\,(1 + k/m)\,(k/k+m)^{\gamma}\,, \gamma \in [0, 1/2)$$

The Proposed Algorithm

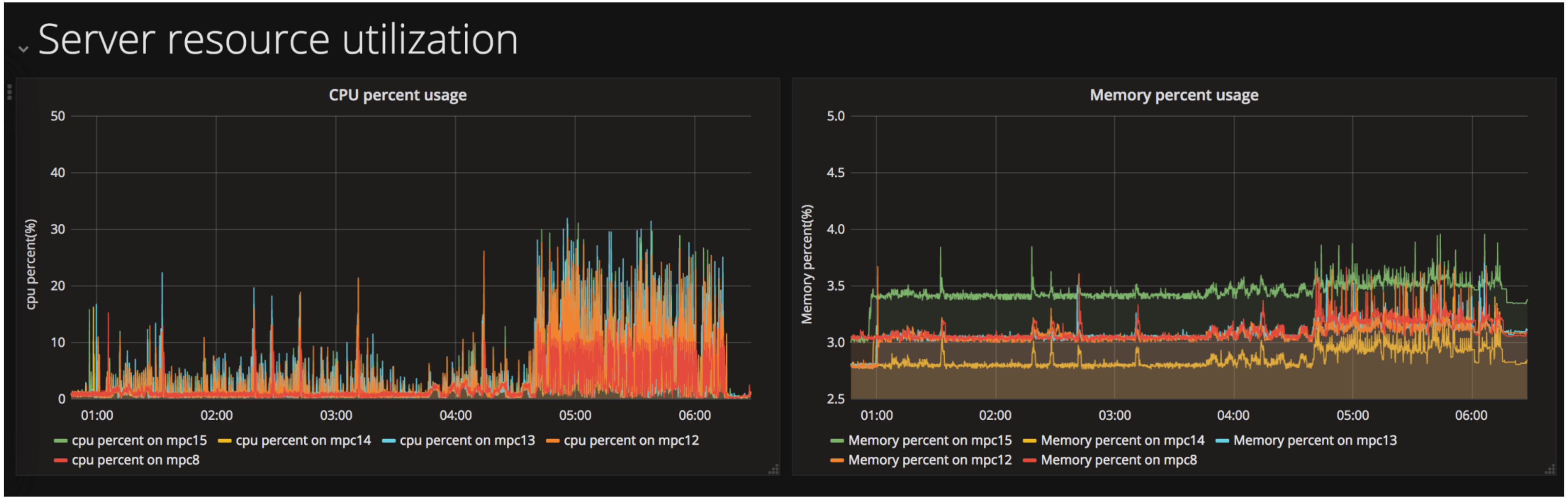
- Assume that monitoring period starts at $t = m.s.$
- **Step 1:** Compute the retrospective statistic $M_{h,} h = 1,\ldots.,m.s$ in combination with our segmentation algorithm for the whole historical period,
  - if no changes are detected the training sample is m = m.s,
  - else the training sample is $m = [cp_{last}, m,s]$, where cp_{last} is the last detected time of change.
- **Step 2:** Compute the sequential statistic for the given time space $k = [m,s,w],$
  - if a change is detected at time $t = ocp$ apply $Tl\ (ocp)$ to estimate the direction of change and deploy or displace a Unikernel.
    - Return to **Step 1** and the monitoring period starts at $t = ocp + d,$ where d the time period in which we assume that a change cannot be occurred,
  - else if no change is detected in the time space $k = [m,s,w],$
    - Return to **Step 1** and the monitoring period starts *at t = w.*

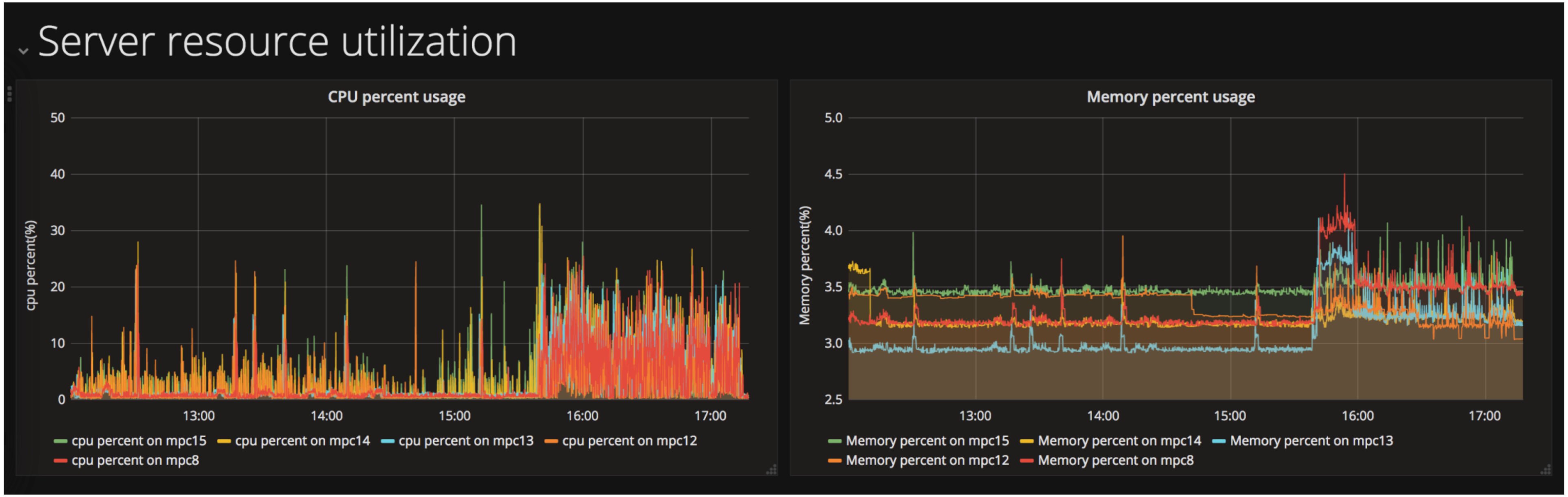
Note: We maintain two parallel change-point detection processes with different significance level and parameter *γ values*. We place one more Unikernel, in case the change is detected from both processes, otherwise we may remove one or two, in a similar way.

Example Detection of Changes

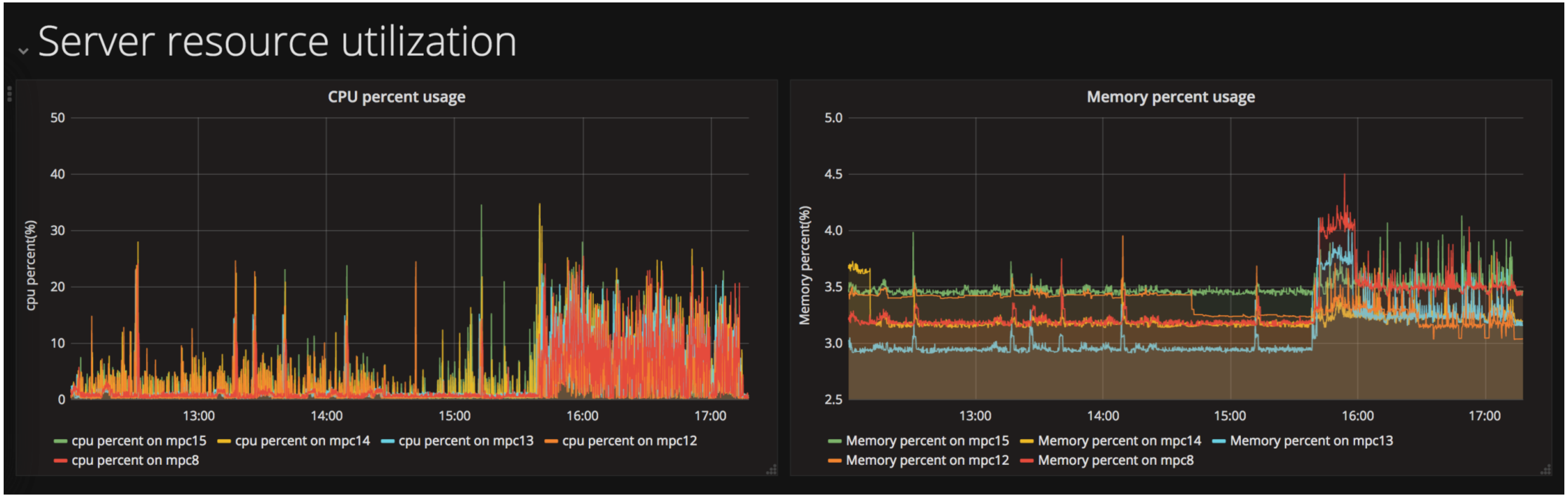


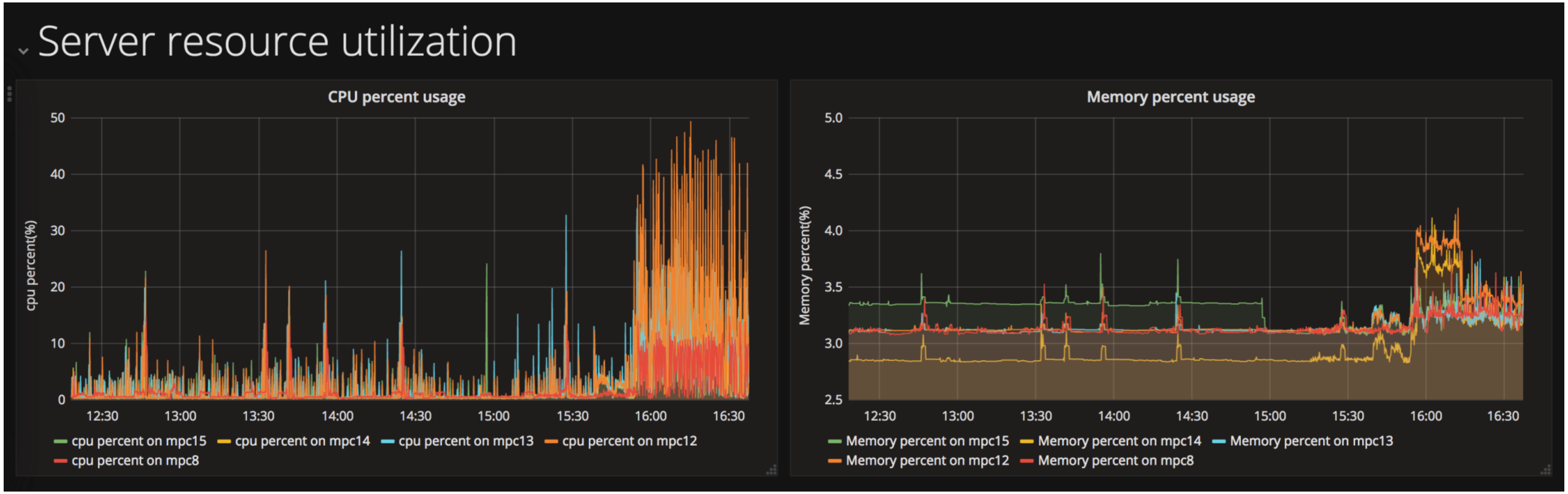
Impact of Change-point Detection Analysis





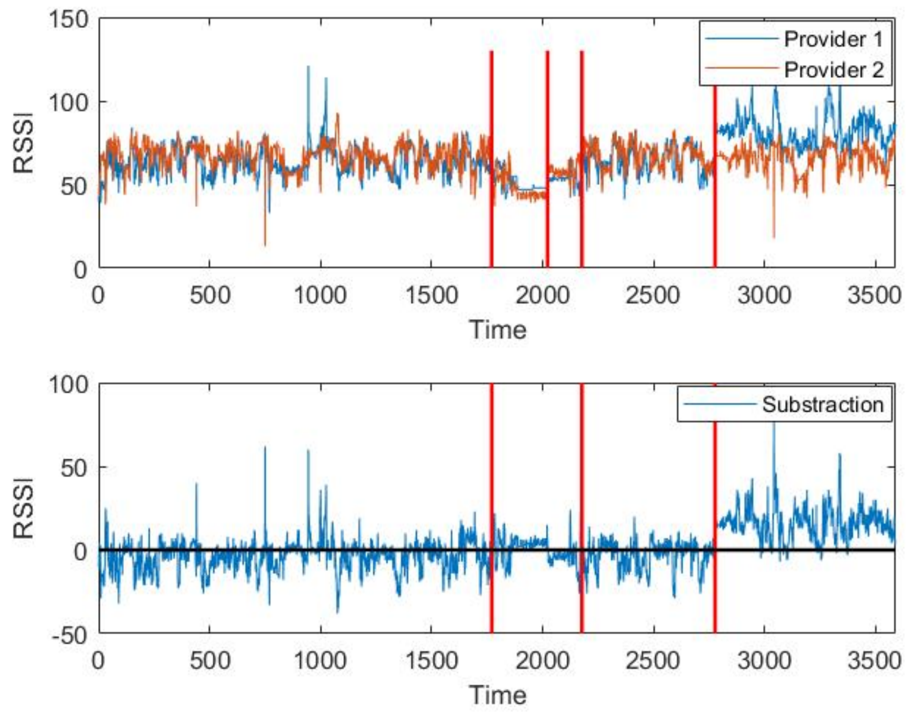
Impact of VM Placement Algorithm



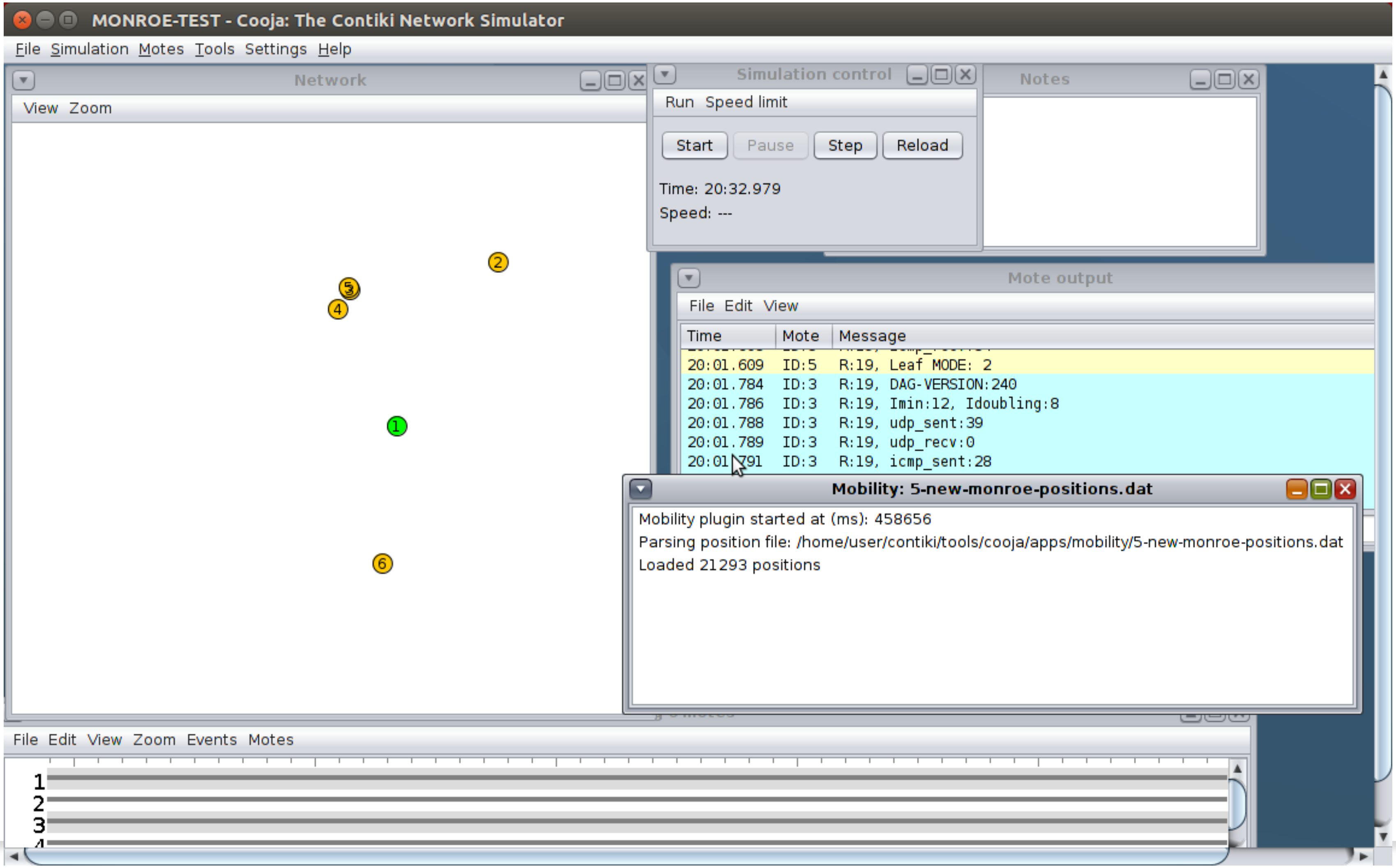


Multi-Homing Capability

- We plan to combine change-point detection analysis with stochastic models
  - guiding the choice of provider to use
  - forecasting the characteristics of upcoming change points



Synthetic IoT Mobility based on MONROE measurements



Related Works

- Indicative relevant platforms:
  - The MOSTO platform [3] deploys Unikernels as TCP proxies to improve TCP's slow-start algorithm performance.
  - A CDN solution with impressive performance [4] provides content through Click OS Unikernels and is evaluated with a CDN simulator.

- In contrast, **MEC with MONROE as the main enabler**:
  - considers **Mobile-Edge/Fog Computing** contexts
  - supports **intelligent multi-homing** capabilities
  - is evaluated in a **real mobile environment**
  - focuses on **flexibility** through:
    - unikernel-oriented VM placement driven by novel content popularity change detection.

Conclusions

- Our general goal is to bring together mobile environments with unikernel-based cloud resources and investigate the novel associated research problems

- Our theoretical approach calls for further improvements, e.g.,:
  - a better way to manipulate the Unikernels' based on the magnitude of change
  - to combine the change-point analysis with stochastic models to implement prediction

- The platform modularity enables real experimentation with a number of theoretical approaches beyond content popularity detection, such us on:
  - radio conditions forecasting;
  - traffic analysis;
  - efficient resource allocation ;
  - network slicing, etc.

Related Publications

1. P. Valsamas, S. Skaperas, and L. Mamatas, “Elastic Content Distribution based on Unikernels and Change-Point Analysis”, European Wireless 2018, Catania, Italy, submitted.
2. S. Skaperas and L. Mamatas, “Change point detection analysis for load balancing based on content popularity”, Mathematics of Networks 2017, Sussex. UK.
3. T. Theodorou, and L. Mamatas, "Software Defined Topology Control Strategies for the Internet of Things”, IEEE NFV-SDN'17, O4SDI Workshop, Berlin, 2017.
4. T. Theodorou, G. Violettas and L. Mamatas, “Softwarized Internet of Things with Lightweight Clouds in Practice”, IEEE NFV-SDN'17, Berlin, 2017 (Tutorial).
5. T. Theodorou, and L. Mamatas, "CORAL-SDN: A Software-Defined Networking Solution for the Internet of Things”, NFV-SDN'17, Berlin, 2017 (Demo Paper).
6. L. Mamatas, T. Theodorou, G. Violettas, S. Petridou, A. Tsioukas, “Intelligent Network Control for the Internet of Things INTER-IOT”, eWINE Grand Challenge, EuCNC 2017, Oulu, 2017 (1st runner up award).

Thank You

Acknowledgments

This trip and dissemination activity was partially funded from:

GENERAL SECRETARIAT FOR RESEARCH AND TECHNOLOGY

The project titled "GSRT FUNDING FOR THE YEARS 2016-2017 (Award for the participation in competitive E.U. projects) [Measuring Mobile Broadband Networks in Europe (MONROE)]", funded from the Ministry of Education, Research and Religious Affairs, General Secretariat for Research and Technology (GSRT), is a participation award for the project titled "Multihoming with Ephemeral Clouds on the Move (MONROE-MEC), HORIZON 2020, GA -Project Number: 644399: Measuring Mobile Broadband Networks in Europe – MONROE" which was funded from the European Union and implemented in cooperation of the University of Macedonia with partners from other countries – members of the European Union.

Related works

[1] A. Madhavapeddy, R. Mortier, C. Rotsos, D. Scott, B. Singh, T. Gazagnaire, S. Smith, S. Hand, and J. Crowcroft, "Unikernels: Library operating systems for the cloud," in ACM SIGPLAN Notices, 2013, vol. 48, pp. 461–472.
[2] A. Madhavapeddy, T. Leonard, M. Skjegstad, T. Gazagnaire, D. Sheets, D. Scott, R. Mortier, A. Chaudhry, B. Singh, J. Ludlam, and others, "Jitsu: Just-in-time summoning of unikernels," in 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15), 2015, pp. 559–573.
[3] G. Siracusano, R. Bifulco, M. Trevisan, T. Jacobs, S. Kuenzer, S. Salsano, N. Blefari-Melazzi and F. Huici. "Re-designing Dynamic Content Delivery in the Light of a Virtualized Infrastructure," IEEE Journal on Selected Areas in Communications, vol. 35, is. 11, pp. 2574-2585, 2017.
[4] S. Kuenzer, A. Ivanov, F. Manco, J. Mendes, Y. Volchkov, F. Schmidt, K. Yasukata, M. Honda and F. Huici, "Unikernels Everywhere: The case for Elastic CDNs," in Proceedings of the 13th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE '17). ACM, New York, NY, USA, 2017.
[5] Manco, Filipe, et al. "My VM is Lighter (and Safer) than your Container." Proceedings of the 26th Symposium on Operating Systems Principles. ACM, 2017.
[6] Mirage OS, http://www.openmirage.org/