Contestion Aware Web of Things Emulation Testbed

R. Hashemian
University of Calgary

D. Krishnamurthy
University of Calgary

N. Carlsson
LinkÖpings University

M. Arlitt
University of Calgary

11th ACM/SPEC International Conference on Performance Engineering
ICPE2020
April 24, 2020

Raoufeh Hashemian | Software Engineer at Cisco Systems | rahashem@cisco.com

- This work was done as part of my PhD research at University of Calgary
Problem

- Web of Things
  - HTTP, CoAP (Constrained Application Protocol - RFC 7252)
- Real testbeds are not always available
- Emulation tools:
  - Use multicore systems to emulate large number of WoT devices
  - Consider effect of resource contention on test results

Developing a scalable WoT emulation testbed while monitoring impact of resource contention on the test results
Objectives

A Web of Things environment with CoAP enabled devices

1. Emulate a large number of devices
2. Evaluate impact of request arrival pattern
3. Compare application layer configurations
4. Consider network characteristics
Solution

• **WoTbench**: Web of Things benchmark

![Diagram of WoTbench components]

- Host: Multicore Linux Machine
  - Test harness
  - Synthetic trace generator
  - Resource monitoring module
  - Reporting module

- WoTbench Core

- Virtual Bridge network

- Gateway Emulator

- Pumba Docker

- Network emulator container

- Devices
  - Device # 1
    - WoT-device
  - Device # n
    - WoT-device

- Connect to other machines hosting devices

- Container

- Component

- Local process

- Test data flow
Emulated WoT-Device

- Uses libCoAP library (https://libcoap.net/)
- List of resources (sensors/actors)
  - Configurable service time specification

<table>
<thead>
<tr>
<th>Resource name</th>
<th>service time (distribution)</th>
<th>Busy/Sleep</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED Switch</td>
<td>20 msec - e</td>
<td>S</td>
</tr>
<tr>
<td>Temp Sensor</td>
<td>5 msec - d</td>
<td>S</td>
</tr>
</tbody>
</table>

- Running on Docker containers
  - Lightweight
  - WoT-device can be replaced by custom applications that support CoAP
Gateway Emulator

- Role of a load generator in conventional Web benchmarking tools
- Use realistic access traces or a trace generated by synthetic trace generator
- Async/single threaded but can use multiple instances
Gateway Emulator

Workload Accuracy

Request Inter Arrival Time (usec)

P[X<x]
Deployment

- Example of deployment decisions: use of processor affinity
- Device capability can be emulated by CPU share
Contestion Issue

- Effect of contention for shared resources on test results

- Test infrastructure lacks capacity to emulate this scenario.
Contention Detection Module

- An instrumented, lightweight WoT-device → CD node
- An extra Gateway Emulator
- A controlled workload

Deployment with CD module
Contention Detection Module

- CD workload:
  - Deterministic inter-arrival time
  - Deterministic service time

- CD-node response time starts to increase at the same throughput that the response time of actual devices increase.
Contetion Detection Module

Response Time Deviations = \frac{\text{Response time} - \text{Service time}}{\text{Service time}}

- \text{CD-node can follow the response time of WoT-devices for both CPU and non-CPU intensive workloads}
Summary

- WoTbench is designed to be deployed on commodity multicore hardware.
- Use cases are capacity planning, testing protocol configuration and effect of network characteristics.
- Contention in shared resources of multicore machine can impact emulation results.
- Contention Detection module is designed to detect such effect and approve/reject test results.
- Future work will focus on auto deployment techniques for WoTbench.
Thank you!

Raoufeh Hashemian  rahashem@cisco.com
Diwakar Krishnamurthy dkrishna@ucalgary.ca

• This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.