Medical Devices Working Group Update

Kate Stewart, Chair
Milan Lakhani, Vice-chair

Jan 17, 2024

Aerospace · Automotive · Linux Features
Medical Devices · OS Engineering Process
Safety Architecture · Systems · Tools

Background

Use case for investigating Linux as a component in a system.

- First workshop identified OpenAPS as good candidate
  - Sources were available to hobbyist community, no NDA required
  - Community willing to engage and explain
  - Started analysis of openAPS project
- Group decided to add in analysis on Open Source Medical Ventilator
- Renamed working group to “Medical Devices”
- Investigation on 62304 requirements pertaining to SOUP for openAPS
- Steady progress working on STPA analysis of OpenAPS over time.

IEC 62304

Comparison of results of STPA analysis to 62304 Software of Unknown Provenance (SOUP) was raised during one of the 2021 workshops.

Report on “IEC 62304 requirements pertaining to SOUP” for OpenAPS project and has been moved to github repository:
https://github.com/elisa-tech/wg-medical-devices/blob/main/62304-soup/main.md

STPA Approach

“STPA (System-Theoretic Process Analysis) is a relatively new hazard analysis technique based on an extended model of accident causation.”

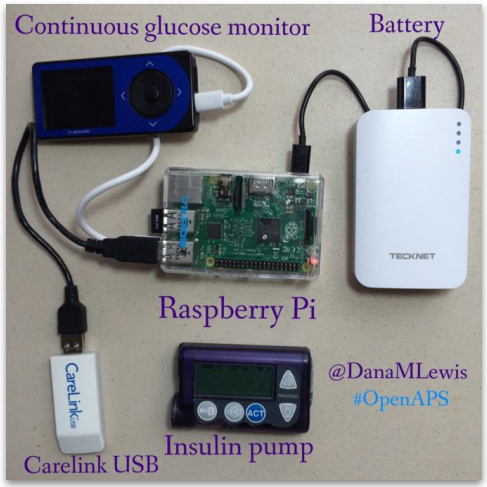
Handbook: http://psas.scripts.mit.edu/home/get_file.php?name=STPA_handbook.pdf

Why?

- Takes an iterative approach
- Diagrams and discussions - makes intuitive sense
- Handbook available to guide us, some expertise in methodology available
- Can take analysis all the way down to linux syscalls & interfaces

Why Study OpenAPS? Community & Open

- https://openaps.org/
- https://github.com/openaps

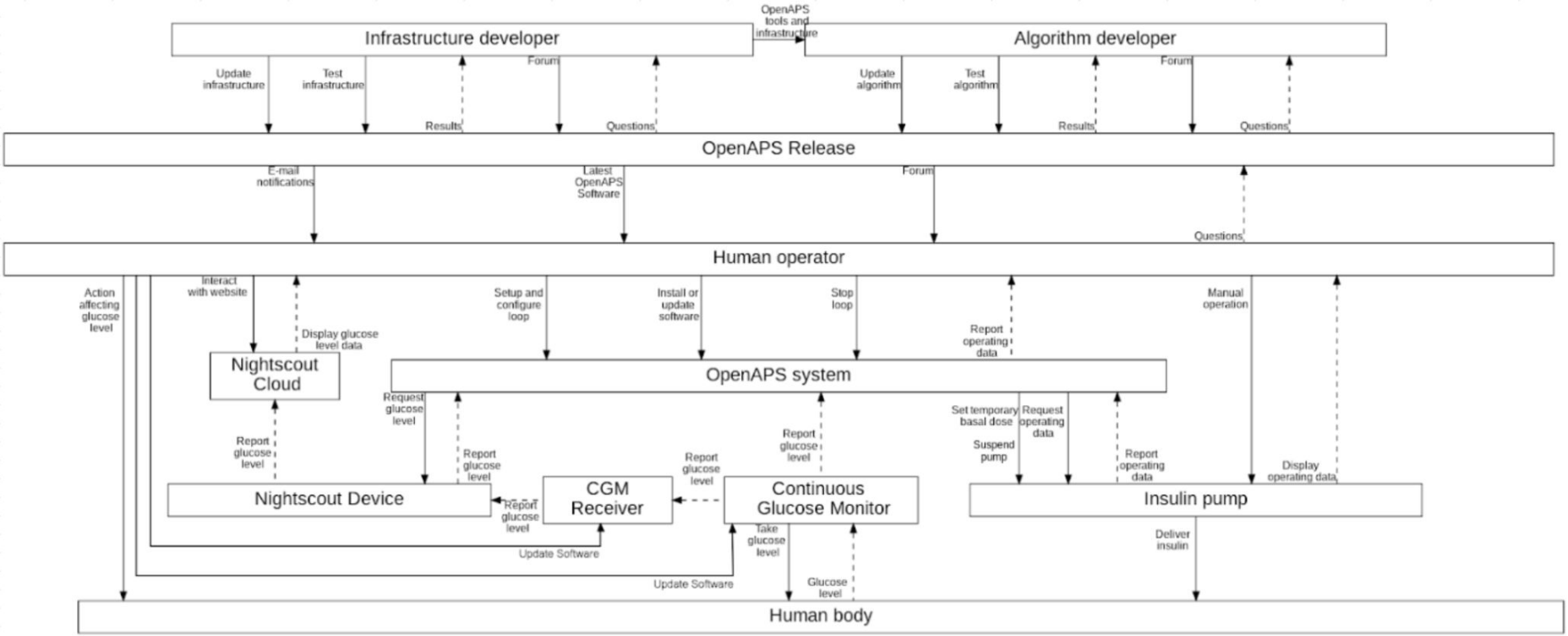


Source: https://diyps.org/2016/09/15/openaps-rigs-are-shrinking-in-size/

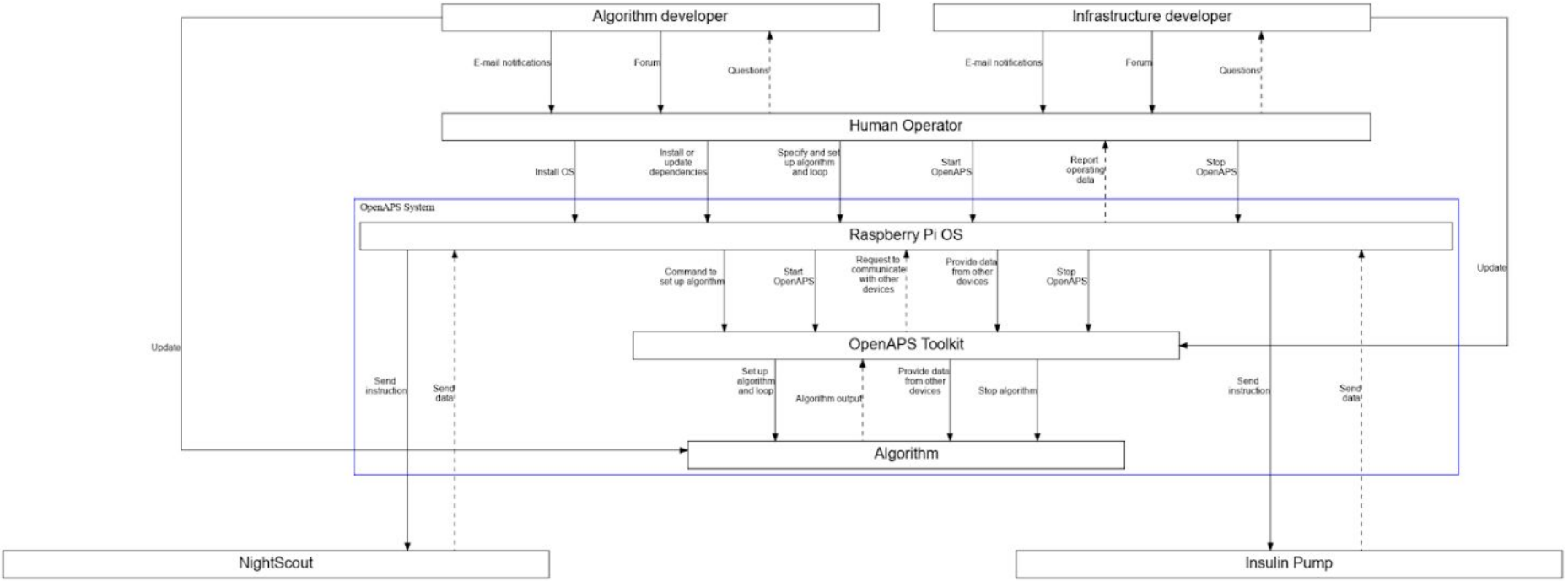
Source: blood glucose graph picture was a screenshot from https://www.youtube.com/watch?v=p76hGxv3-HE

More research studies at:
https://www2.diabetes.org/newsroom/press-releases/2022/new-study-shows-open-sourced-autmoated-insulin-delivery-safe-effective-treatment-option-type-1

L1 Control Diagram: Interaction with Environment



L2 Control Diagram: Focus in on OpenAPS



System views

Static System View
- Supported system calls
- Static modules
- Dynamic modules
- Kernel Config options

Dynamic System View
- System calls invoked
- ioctls invoked
- Subsystems use

Tracing openAPS workload

Thank you to OpenAPS community for providing real workload traces!

- **Methodology**
  - Discover Linux kernel subsystems used by openAPS
  - Enable event tracing before starting the workload.
  - Extract system call numbers from trace and map them to system calls
    - Collect supported system calls using auditd package tool: ausyscall --dump
  - Trace openAPS application (kernel tracing & strace)
  - Gather static and dynamic module information
- **Tools employed**
  - ausyscall --dump
  - Kernel tracing
  - Lsmod
  - scripts/checksyscalls.sh (Linux kernel script)

Methodology upstreamed to: https://docs.kernel.org/admin-guide/workload-tracing.html

2023 Medical Devices Update

- Completed iterations of 1st and 2nd level analysis based on feedback to date.
  - Had to iterate and refine top level diagram a couple of times.
  - Refined Loss Scenario, UCA analysis & Requirements for L1 & L2.
  - Published results in github repository after converting from spreadsheet

- Developed tool to convert STPA analysis spreadsheet requirements into machine readable YAML format.
  - Published tool in github. https://github.com/elisa-tech/wg-medical-devices/tree/main/applying-stpa

2023 Medical Devices Update

- Developed methodology
  - worked with OpenAPS community to get traces
  - Trace linux kernel and identify key API & components for L3; partial → architecture of kernel.
  - https://github.com/elisa-tech/ELISA-White-Papers/blob/master/Processes/Discovering_Linux_kernel_subsystems_used_by_a_workload.md
  - Methodology upstreamed to: https://docs.kernel.org/admin-guide/workload-tracing.html

- OpenAPS running on a virtual machine. (Thanks for help from Tooling)

Work Planned for 2024…

<table>
<tr><td rowspan="2">Applications</td><td>Applications</td><td>Applications</td></tr>
<tr><td></td><td>Runtimes (Python, Json)</td></tr>
<tr><td></td><td colspan="2">System Libraries (watchdog?)</td></tr>
<tr><td colspan="3">System Call Interface (32-bit libraries)</td></tr>
<tr><td>VFS</td><td>Sockets</td><td rowspan="2">Scheduler
- Data request every 5 minutes
- Time drivers used for setting a watchdog?</td></tr>
<tr><td>File Systems</td><td>TCP/UDP</td></tr>
<tr><td>Volume Manager</td><td>IP</td><td rowspan="2">Virtual Memory

Timer</td></tr>
<tr><td>Block Device</td><td>Net Device (802.11)</td></tr>
<tr><td colspan="3">Device Drivers (Raspi Spidev, dwc otg, BGPIO, BT,...)</td></tr>
</table>

- Discuss with Architecture Working Group how we want to tackle the L3 analysis with common parts of the kernel.
- Moving OpenAPS to build with Yocto, and generating SBOM
- Clean up overview: White paper of the L1 & L2 analysis for OpenAPS
- Looking for other open medical device application to compare & contrast

ELISA
ENABLING LINUX IN SAFETY APPLICATIONS

Aerospace · Automotive · Linux Features · Medical Devices
OS Engineering Process · Safety Architecture · Systems · Tools

Interested to help:

https://lists.elisa.tech/g/medical-devices

Licensing of Project Update Results

All work created during the workshop is licensed under *Creative Commons Attribution 4.0 International (CC-BY-4.0)* [https://creativecommons.org/licenses/by/4.0/] by default, or under another suitable open-source license, e.g., GPL-2.0 for kernel code contributions.

You are free to:

- **Share** — copy and redistribute the material in any medium or format
- **Adapt** — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.