Xen based Arm Autonomy Stack using Yocto

Linaro Tech Days 2020

Bertrand Marquis, Jon Mason Arm Ltd
March 24th 2020
Presenters

• Bertrand Marquis
  • Principal software engineer at Arm ltd in the Open Source Software group (CE-OSS)
    – Yocto Autonomy Reference Stack
    – Xen on Arm
  • Bertrand.marquis@arm.com

• Jon Mason
  • Principal Yocto Developer at Arm ltd in the Open Source Software group (CE-OSS)
    – Principal maintainer of the meta-arm OpenEmbedded Layer
    – OpenEmbedded Board member
  • jon.mason@arm.com
Summary

• Xen based Arm Autonomy Reference Stack
  • Why and what?
• meta-arm Yocto Layers
  • How?
• Current Status and Future
Hypervisor enabled reference stack

• Autonomy systems are complex
  • Lots of functions
  • Enough computing power

• Hypervisor advantages
  • Easier development and reuse
    – Group common, non-differentiating features into a separate project
    – 3rd parties can focus on value delivery
    – Platform invariant applications
    – Sets a starting point for code reuse at vendor/T1 level
  • Easier deployment
    – Isolate critical from non-critical
    – Control boot flow
    – Easier updates
Why Xen

• Xen is a type 1 hypervisor
  • No need to certify Linux or windows
  • Reduced size of code
  • Fast boot time

• Xen is open-source
  • Widely available
  • Strong community
  • Used in number of servers and systems

• Xen community is pushing for this kind of applications
  • FuSa project for Xen certification
  • Automotive/Embedded is a Xen goal
  • Some Industry leaders implicated (Qualcomm, Xilinx, BAE, Bosh, Arm, ...etc)
Autonomy reference stack

• Build a Xen based reference stack
  • Suitable for autonomy systems
    – Autonomous driving, robotics,
    – Suitable for safety and/or real-time
  • Focus on boot time and stripped functionalities
  • Small and simple system to create/deploy/start/stop guests
  • Functions to provide safety and security features
    – OP-TEE integration
    – Secure boot
    – Update system
  • Allow easy integration of different components
    – Have guests useable on different systems independently of the hardware
Autonomy reference stack

- Dom0: Xen guest Manager, Workload, Linux Kernel
- DomD: Workload, Kernel/RTOS
- DomU: Workload, Kernel/RTOS
- DomU: Workload, Kernel/RTOS

Autonomy Reference Stack
External component
Hardware - SoC
Standards based interface

OP-TEE

TF-A / U-boot

SoC
How do we build a custom reference stack?

Yocto!

- The Yocto Project is a set of templates, tools and methods that helps you build custom Linux-based systems.
What is an OpenEmbedded/Yocto Layer?

Custom Linux Distribution

- meta-arm-bsp
- meta-arm
- meta-oe
- meta-browser
- meta-rust
- meta-clang

Custom Linux Distribution

- meta-xilinx/ti/rockchip
- meta-arm
- meta-oe
- meta-browser
- meta-rust
- meta-clang
Why meta-arm collection of layers

• Many Arm vendors have their own unique Yocto meta layer
  • Fragmentation of implementation
    – Divergent recipes
    – Different versions of underlying software
    – No need to upstream (random git tree with their changes on top)
  • Difficult to upgrade to new versions of Yocto and the underlying software (stale)
  • No shared work to maintain the recipes and underlying software
    – And often not thoroughly tested
  • Often not following Arm recommended processes or implementation
    – I.e., not using OP-TEE or secure boot
meta-arm

• Intended to be the "one stop shop" for all Arm Software
  • Arm specific software
  • Reference hardware/BSP enablement
  • Arm Toolchains
  • Autonomy/Automotive
  • Arm Embedded Distro?

• Also intended on being a place to collaborate

• CI constantly run to ensure stability and quality

• Model the behavior for Arm platforms that we want vendors to emulate (and hopefully just copy)
meta-arm

- Name makes it a bit confusing, meta-arm/meta-arm
- This layer provides support for general recipes for the Arm architecture. Anything that's not needed explicitly for BSPs, a distribution, or destined to be upstreamed to another meta layer goes here
- Currently contains recipes for:
  - OP-TEE
  - TF-A
  - OpenSCD
meta-arm-bsp

• This layer provides support for Arm reference platforms
• Currently contains support for:
  • Cortex A5 DesignStart
  • GEM5
  • Juno
  • Foundation Models
meta-arm-toolchain

• This layer provides support for Arm's GNU-A toolset releases
• Currently contains support for Arm modified GCC v8.2, v8.3, and v9.2
• Pre-compiled binary toolchains also available for use via this layer
meta-arm-autonomy: Goals

- Autonomy reference stack implementation based on Yocto
- Build all components
  - Host project (Dom0)
  - Guest project (DomU or DomD)
  - Xen
  - Firmware (TF-A, OP-TEE)
- Be useable with other layers
  - Use external BSPs
  - Use existing workloads for guest (AGL, wayland or others)
- Easy to use for fast prototyping and research
meta-arm-autonomy: Host project

• Build a host system:
  • Build Xen (using meta-virtualization)
  • Build a device tree to boot Xen and Dom0
    – Add xen bootargs
    – Add Dom0 multiboot node
  • Build Dom0 Linux
    – Add Xen backend drivers in Linux kernel
    – Xen Dom0 tools
    – Qemu
    – Ssh and networking

• Implemented as a Yocto distribution feature:
  • DISTRO_FEATURES += "arm-autonomy-host"
  • arm-autonomy-host-image-minimal Yocto image packs dom0 minimal components
meta-arm-autonomy: Guest project

• Implemented as a Yocto distribution feature:
  • DISTRO_FEATURES += "arm-autonomy-guest"
  • Customize an image to be used as Xen DomU
    – Xen frontend drivers
    – Console on HVC0
  • To be used with any Yocto image (no specific image provided)

• Arm64-autonomy-guest BSP
  • Minimal Kernel configuration (no hardware drivers) to run as Xen DomU
  • Enable arm-autonomy-guest DISTRO_FEATURE
    – if used no need to modify DISTRO_FEATURES
  • Suitable for DomUs only relying on Xen drivers for communication
meta-arm-autonomy: Xenguest System

• Create and manage Xen guests:
  • Xenguest image format:
    – Include all components in one file
    – Xenguest-mkimage tool
      ▪ create/modify/check images
      ▪ useable on development host and on target
  • Xenguest manager
    – Dom0 tool to manage images
    – Create guest from images
      ▪ Create disks using LVM
    – Start/stop guests
      ▪ Create final xen configuration
      ▪ Call init scripts at different steps

Xenguest Image

- Linux kernel
- Device tree
- Xen config
- Disk definition
- Disk content
- Init scripts
meta-arm-autonomy: Xenguest System

• Integration in meta-arm-autonomy:
  • Xenguest image FSTYPE
    – Include rootfs, kernel, xen configuration and init scripts in an image
    – Activated by "arm-autonomy-guest" distro feature
    – Variables can be used to customize it (size of ram, number of cores, command line, etc.)
  • Xenguest-image-extra class to extend the image from other recipes
  • Xenguest-manager
    – Included in "arm-autonomy-host-image-minimal"
    – Init script to boot automatically guests on startup
    – Variables can be used to customize it (disk for LVM volumes, path to guests to include, etc.)
  • Xenguest-network-bridge
    – Create a network bridge on host and connect guests to it
    – Parameter in xenguest image to connect it or not to the bridge
arm

Current status and future
meta-arm Status

• Initial deployment in January 2020

• Initial feedback from Arm Licensees is promising

• Beta of meta-arm-autonomy has been pushed to the mailing list last week !!

• First version to be released with Yocto Dunfell in April
meta-arm future

• More BSPs!
  • Corstone-700
  • Neoverse N1 SDP
  • (more after Q1 2020)

• More Software!
  • EDK2 (UEFI)
  • SCP
  • TF-M

• Enhance meta-arm-autonomy!
  • Boot time and size optimization
  • Security and Updates
  • Easier development
  • Compatibility with other layers (BSPs or workloads)
How to get involved

• More information on Yocto Project
  • https://www.yoctoproject.org/docs/

• Clone and use the layer for all of your Yocto based Arm projects
  • git://git.yoctoproject.org/meta-arm

• Join the meta-arm mailing list
  • https://lists.yoctoproject.org/g/meta-arm
Thank You
Danke
Merci
Merci
谢谢
ありがとう
Gracias
Kiitos
감사합니다
धन्यवाद
شكرًا
ধন্যবাদ
תודה