WIND™

Enterprise Linux vs. Embedded Linux

WHEN IT MATTERS, IT RUNS ON WIND RIVER

EXECUTIVE SUMMARY

The use of open source operating systems has garnered more attention in recent years for a variety of market segments and usages. In part, the initial low cost of development, as well as a growing number of use cases, has driven this increase in popularity. Linux is one of the most widely used open source operating systems, but there are numerous different Linux offerings. This wide set of choices can make it confusing and challenging to determine the right version for a given use case. This paper offers an introduction to Linux for the enterprise and Linux for embedded products.

TABLE OF CONTENTS

Executive Summary 2
Enterprise vs. Embedded Linux 3
Enterprise Linux 3
Hybrid Enterprise/Embedded Linux 4
Embedded Linux 4
Wind River Linux for Embedded Devices 5
Conclusion 5

WIND

ENTERPRISE VS. EMBEDDED LINUX

The overall Linux landscape can be broken into two device target segments: enterprise and embedded. As we continue to see shifts toward a blurring of the information technology (IT) and operational technology (OT) market segments, these device segments should not be considered unique groups but part of a continuum of overlapping features and requirements. For example, enterprise and embedded developers alike share the desire for both a predefined binary system and a fully source code–based system. One of the benefits of the Linux ecosystem is that one size does not have to fit all, as there are many options to customize a system for a particular market.

The traditional enterprise Linux market is designed to accommodate better-resourced, multiuser, multipurpose systems. Generally, IT administrators and users manage and use the content for these systems, as opposed to operating system developers and engineers. With an enterprise distribution, the operating system provider generally defines the content, configuring the components to provide a "one size fits most" solution. Enterprise distribution providers usually make available a very large range of pre-compiled software, but this software is not specifically tailored for any one solution. They typically include source code for all components as required by their various open source or proprietary licenses, but it is not intended to be used by anyone to change the way in which the "one size fits most" solution works. To help improve the customer experience, enterprise Linux distributors limit customizations to allow for package installation and editing of documented configuration files. By limiting customization, they make it possible to provide low-cost support. Since most users have similar systems and will experience similar usage caveats and defects, this helps enterprise distribution providers leverage economies of scale where any identified issues will be resolved and ultimately benefit all current and future users.

The lifecycle of enterprise Linux usually runs approximately three to five years. This relatively shorter life for enterprise Linux software applications is due to IT use cases requiring shorter lifecycles to maximize data center performance, as well as to the costs associated with maintaining security and bug fixes that can arise over longer lifecycles. For embedded Linux applications, the lifecycles are usually much longer because many embedded systems are built into critical infrastructure, where the economics of customization, reliability, and other design requirements drive higher costs that need to be spread over a longer lifecycle.

Traditionally, embedded systems are special purpose systems, often with specific capabilities and no direct end user interaction. Underlying operating systems are generally cross-compiled; software development takes place in a more highly resourced environment, such as a desktop, and is then tested and distributed on a lesser-resourced target device. Similar to traditional embedded systems, embedded Linux systems differ from enterprise Linux in large part because system design constraints resulting from limited hardware and memory necessitate customizations and building software from source code.

These variances often manifest in significant differences in the operating system support model. For example, a commercial ecosystem for embedded Linux requires not only a supported target runtime environment to deploy and test the distribution but also a supported build environment to create, develop, customize, and maintain the runtime. The scale of configurations that require support is dramatically different for embedded Linux. While many of the development and configuration problems may be similar, the scope and breadth of the problems and number of configurations is much larger than the enterprise space. While some cost savings can be found through economies of scale, this matrix of various target hardware capabilities and development environments increases the support requirements and costs associated with defects and maintenance support.

The continuum of enterprise Linux and embedded Linux can be further summarized into three main categories: enterprise Linux, hybrid enterprise/embedded Linux, and embedded Linux. Each of these categories has unique usage, design, and support traits that may influence which is best for any given project.

ENTERPRISE LINUX

Often when people think of Linux operating systems, they are considering the enterprise Linux offerings from Red Hat, Ubuntu, and others. These are truly general purpose operating systems, as they are not built for a single purpose use for a specified device. The operating system vendor handles the design and compilation of the operating system environment. The user is limited to

LINUX SYSTEMS FOR LARGE AND SMALL DEVICES

There are many popular GNU/Linux distributions, including Arch Linux, CentOS, Debian, Fedora, Gentoo, SUSE, and Ubuntu. For larger devices, such as mainframe or supercomputers, there are both standard and custom Linux distributions available. For smaller devices, special purpose components and environments provide alternatives to the GNU operating system components. This is especially true of embedded systems, which often use specially tailored systems that may—or may not—be derived from the GNU environment. The most common of these non-GNU–derived systems is the Android operating system that serves as the heart of many smartphones. If you include both smartphones and embedded systems, Linux has the largest installed base of all general-purpose operating systems worldwide.

selecting a configuration or edition, installing additional software packages, and handling the administrative configuration of individual items. Everything is expected to be used in a purely binary environment by IT staff and general users. Distributors often include long-term support with the caveat that they only support certain common components, such as unmodified software and common usage considered typical for the platform targeted toward use cases with relatively shorter product lifecycles, such as three to five years.

There are special purpose enterprise systems designed to run general purpose components. Both Project Atomic and Ubuntu Core use a small subset of the standard components from their base enterprise operating systems. These base components provide just enough of an environment for an IT administrator to deploy other special purpose software packages or containers. Often these containers are designed around providing unique services for system users. While the core environment is special purpose, the ability to extend the system using containers provides a general purpose behavior.

HYBRID ENTERPRISE/EMBEDDED LINUX

Somewhere on the spectrum between enterprise and embedded Linux sits roll-your-own, or customized enterprise distributions. These distributions are still a fairly common way to build a special purpose enterprise or embedded Linux system. Often they are based on a fork of a community-developed distribution. This gives users the mistaken impression that there is a community they can ask for help, when in reality the creator of this custom fork cannot rely on the initial community-created distribution it was derived from. This is mainly because the community had no involvement in the creation, customization, or use of this specific fork of the community distribution. Initially, this support burden can be satisfied by a dedicated expert who can manage the small number of required updates and defect fixes that are part of providing up-to-date distributions. Over time, however, the level of customization and the number of different versions will grow to a point of complexity that will require a significant team of experts to provide maintenance and support, driving increased costs. Since everything is custom, there is little to no benefit from economy of scale to help support this type of system.

EMBEDDED LINUX

Embedded Linux systems power devices people come in contact with on a daily basis but do not typically associate with Linux or Unix. From television sets to smart appliances to routers and cell phones, embedded systems support a wide range of hardware devices for many specialized purposes. There are many well-known embedded Linux systems available to developers. Buildroot is one example. Designed as a simple way for people to develop embedded systems, Buildroot uses existing technologies familiar to most Linux developers, such as kconfig and make, to produce a cross-compiled operating system. In order to keep it simple, the default environment is minimalistic and easy to understand. To build large projects, developers generally have to fork and modify the system. Many of the maintenance problems of a roll-your-own approach are then introduced, as the community is unable to help resolve

issues specific to modifications. Buildroot is often used for the simple systems developed for hardware validation. However, it can also be used on more complex projects such as OpenWRT, which is popular on wireless access points.

Another embedded Linux system is known as both OpenEmbedded and the Yocto Project, as they share a common software core. In reality, OpenEmbedded represents the general open source community, whereas the Yocto Project represents related content, such as training materials and marketing content. Both communities work together, with all of the software components being developed within the OpenEmbedded project. The OpenEmbedded project and Yocto Project use a build system that is designed such that only the extensions themselves require local support. This enables an ecosystem of both community and commercial vendors who can support all of the system's common components.

The Yocto Project is a more robust system than projects such as Buildroot. As a result of this versatility, a potentially steeper learning curve may result, making initial adoption, at least, somewhat more difficult. This learning curve is mitigated by the Yocto Project's advocacy, training materials, and documentation. In addition, the Yocto Project's supported ecosystem is designed to fill in specific gaps that an end user may have. The Yocto Project, due to its versatility and design, is in use across a large number of vertical markets that include devices with specifications and use cases perhaps better described as enterprise in nature.

WIND RIVER LINUX FOR EMBEDDED DEVICES

Wind River® Linux is a Yocto Project Compatible open source baseline and is the embedded operating system of choice for IIoT software developers who want a combination of open source flexibility, commercial grade reliability, and support to help minimize the total cost of ownership. Wind River Linux delivers vital components for the productization and commercialization of any IoT device. Within a VxWorks workload consolidation solution, a Wind River Linux VM can be established to run Linux applications, such as communications.

CONCLUSION

There is a wide range of choices when choosing a Linux operating system, but the various options can be divided into two target segments: enterprise and embedded. Yet, as the environment shifts toward a blending of the IT and OT market segments, feature offerings, development requirements, and marketing messaging will begin to merge and overlap. If a developer or company plans to utilize a Linux open operating system, it is important to evaluate their needs and choose the Linux offering that best meets their requirements.

WIND

Wind River is a global leader in delivering software for IoT. Its technology is found in more than 2 billion devices and is backed by world-class professional services and customer support. Wind River is accelerating digital transformation of critical infrastructure systems that demand the highest levels of safety, security, performance, and reliability.

© 2018 Wind River Systems, Inc. The Wind River logo is a trademark of Wind River Systems, Inc., and Wind River and VxWorks are registered trademarks of Wind River Systems, Inc. Rev. 07/2018