BLACKDUCK

THE ROLE OF OPEN SOURCE IN BUILDING OUT THE INTERNET OF THINGS

The build-out of the Internet of Things (IoT) is advancing at a tremendous rate. By 2020, more than 50 billion intelligent devices are expected to connect to and exchange information over the Internet. This cohort of "things" comprises staggering diversity, from recognizable computers to infrastructure devices to sensors, light switches, and thermostats.

The impact of the IoT will span the gamut of industries and applications, including medical, agriculture, manufacturing, consumer, electronics, transportation, and energy. And like the existing Internet, the emerging IoT will rely upon and promote the adoption of open source technologies and standards. Given the broad range of potential applications and constituents, it's not surprising that disparate visions exist for the evolution of the IoT. While some see the IoT as an incremental extension of existing computing technologies and methods, others view it as a revolution that will reinvent the IT industry, forcing major technology players to rethink their business models radically.

Either way, it's clear that open source software (OSS) will play a major role in the ongoing development of the IoT. This white paper examines the role and reach of OSS in building and sustaining the IoT, from infrastructure and applications to other value-added content. Specifically, it explores how OSS can support competing and complementary architectures and meet looming IoT challenges, including security and privacy.

COMPETING VISIONS FOR THE EVOLUTION OF THE IoT

The commercialization of the Internet of Things, especially of the open source software that will support its build out, will be subject to competing technical and financial models. Currently, there are two prevalent views of the architecture and vision of the IoT. You can think of the two models as *Many Peers* and *Many Leaves*.

With *Many Peers*, the IoT is effectively an extension of the current connected universe. In this paradigm, the IoT comprises a network of "compute peers," deployed with Linux, Android, and comparable high-level operating systems, running on 32- or 64-bit hardware, communicating over existing TCP/IP networks, and extending to applications running on a LAN or in the cloud. *Many Leaves*, by contrast, is an extension of the machine-to-machine paradigm, with a vast cohort of relatively simple end-point systems, deployed with deeply-embedded operating systems, or no operating systems at all, running on a mix of 8-, 16-, and 32-bit hardware, and communicating via specialized interconnects and protocols.

The two visions are not incompatible, and devices implementing both paradigms are already populating the nascent IoT. They differ mainly in terms of who promotes which. *Many Leaves* is the logical favorite of semiconductor suppliers and embedded software vendors, as well as adherents to the maker movement. The *Many Peers* model is embraced by systems vendors and enterprise software suppliers.

OPEN SOURCE BUSINESS MODELS AND THE IoT

There are dozens of ways to monetize open source, and no two companies working in this area are alike in the mix of tactics they employ. Some typical approaches include:

- Using OSS to deliver a service, such as cloud hosting
- Selling services for OSS, such as support for Linux or OpenStack
- Selling products for OSS, such as development tools or training materials
- Offering OSS with commercial upgrades (commercializing an OSS platform or middleware)

At a conceptual level, OSS business models fit into four major categories: building OSS, building with OSS, building for OSS, and building on OSS. Let's briefly examine each as it applies to the Internet of Things.

BUILDING OSS

The most basic (and most challenging) model involves creating and commercializing open source software for direct return. With the Internet of Things, such a model entails building OSS device software (operating systems, middleware or applications, from ISVs large and small), creating IoT-enabling software in the cloud, and developing end-user web and mobile apps – all as open source.

BUILDING WITH OSS

This model applies to hardware and software and is typically the domain of device manufacturers (OEMs) and commercial software providers. IoT leaf nodes and infrastructures deploy open source operating systems, middleware, and other enabling software. The value-added software deployed on these devices, as well as on other nodes across the IoT, may still be proprietary. The ability to create proprietary works with or on top of OSS will of course depend on licensing and software architecture.

BUILDING FOR OSS

Historically, this model has entailed supplying training, documentation, tools, and support for existing open source technologies. With the IoT, there are opportunities to support projects such as OpenRemote or RIOT, offer tools to debug and optimize IoT protocol stacks, and deliver training and education on developing IoT applications with OSS tools and technologies. This model provides ample opportunities for educators, publishers and training companies, as well as integrators and an array of specialized consultancies.

BUILDING ON OSS

This model will, by definition, constitute the default IoT paradigm: all nodes, end-to-end, will have strong dependencies upon OSS and almost any new business involved in the IoT ecosystem will heavily depend on OSS to succeed. In the enterprise, "Building on OSS" traditionally meant running your company with OSS – CRM, accounting, engineering, marketing – any and all business critical operations. With the IoT, this model will entail leveraging the open source aspects of the IoT to run a business. Current examples include energy delivery, industrial automation, premises management, and physical security.

THE IoT EXPANSION: WHERE DOES OSS FIT IN?

While open source will continue to be instrumental to the IoT, its presence and utility are not uniform across all elements of the network.

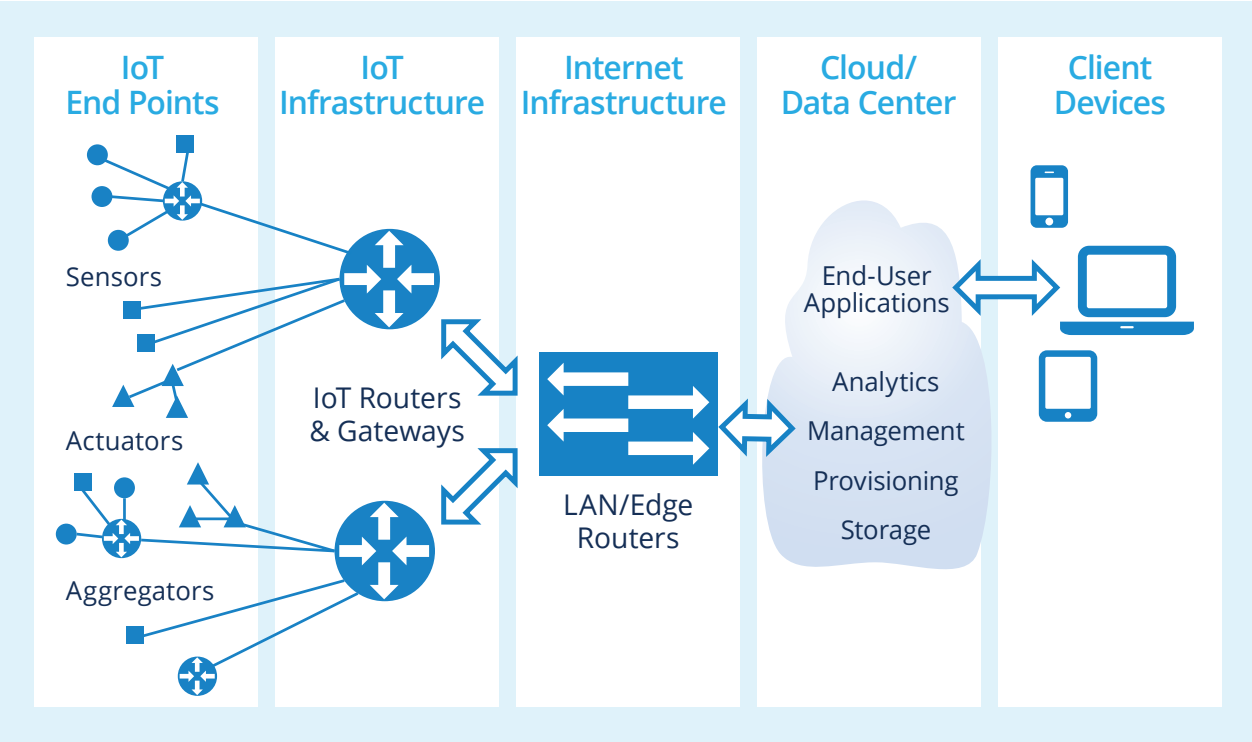

Figure 1. – IoT Node Types and Data Paths

"DUMB DEVICES"

An area often overlooked in IoT discussion is the vast population of "dumb devices" – smart labels, inductive slugs, and other RFID devices, used extensively in manufacturing, inventory control, and other areas to track the location of moderate to high-value items (pharmaceuticals, clothing, etc.). These devices are passive, reporting only an ID and relatively small amounts of data when energized by specialized scanning equipment or when they pass through RF portals (e.g., entering/exiting warehouses). Passive devices can provide a needed bridge to the largest universe of "things."

The role of OSS in such passive devices lies not in the RFID tags and slugs themselves, but in the equipment that energizes and scans them, and in supporting the applications that act upon the data.

SIMPLE END POINTS

The "things" that comprise the leaf nodes of the prototypical Internet of Things are single-function sensor and actuator devices. Such devices are envisioned to be ubiquitous and free-standing, with low power consumption and lower cost. Think light switches and sockets, thermostats and HVAC controls, motion sensors and perimeter alarm switches, soil moisture and air temperature sensors, and so on.

The prototypical leaf node deploys fairly minimal software, supporting core functionality for sensing or affecting its environment and communicating state or status information upstream. Such devices can benefit from an actual embedded operating system or just deploy a main program loop and device service code. The role of OSS in such devices is tactical. Developers will surely use OSS tools to create

leaf node devices, and semiconductor suppliers will provide open source device drivers and other elements to support them, but the applications running on them will likely remain closed. Device manufacturers today (and for the foreseeable future) may see more value in retaining all rights to their differentiating technology, in hardware and software, than they see in sharing development and maintenance responsibilities.

PEER-LEVEL END POINTS

Peer-level leaf devices serve many of the same functions as simple end points, but with two key differences:

1. They are better provisioned, with 32- or even 64-bit CPUs and additional RAM and storage
2. They are more likely to bundle routing and/or gateway functionality into a single package

As such, they are by definition multi-function devices and have the potential (or the necessity) of deploying enterprise-peer operating systems – Linux, BSD, versions of Windows, etc. These devices represent more interesting opportunities for OSS, from system software (especially Linux and Android) up through middleware and applications frameworks, as well as routing software.

IoT INFRASTRUCTURE: THE BASICS

In discussion of open source for the Internet of Things, we should examine two distinct types of infrastructure – routers, gateways, and aggregators that bridge between the existing Internet and IoT end points vs. access points, LAN router and edge routers, backbone and core switches, and routers that constitute the Internet.

At this level, the IoT still greatly resembles its conceptual predecessor, Machine-to-Machine (M2M) networking. Mission-specific devices transmit context-dependent information across a point-to-point or mesh networks, aggregated, buffered, and conditioned by application-specific gateways and routers. In the M2M paradigm, these devices communicated over a LAN to computers tasked with control, data analysis, etc. With the IoT, they bridge to the larger Internet and to cloud servers.

These nodes provide ample opportunities for OSS deployment and for the evolution of new open source implementations.

INTERNET INFRASTRUCTURE

The broader infrastructure of the Internet, from local wireless networks to broadband and mobile baseband access, to edge and core networking, is already teeming with open source software:

- Embedded Linux and Carrier Grade Linux in access points, routers, gateways, firewalls, media gateways, and other networking and telecommunications equipment
- Open source routing packages, security libraries, network management tool kits, high-availability enablers, and other network-related middleware
- BSDLite-derived TCP/IP stacks paired with proprietary embedded operating systems
- Embedded web servers and web application components used to support configuration and management interfaces

The ongoing rollout of SDN (Software Defined Networking) and NFV (Network Functions Virtualization) is also providing ample new opportunities for open source development to support Internet infrastructure.

THE CLOUD

As with Internet infrastructure, the cloud is substantially built on open source software components. Linux, virtualization platforms, orchestration and management software, application support libraries and other types of cloud middleware, and the tools and frameworks developers use to write and deploy code – all open source.

That is not to say that all cloud software is open (e.g., Microsoft Azure), nor that software that implements Infrastructure-as-a-Service (IaaS) and Platform-as-a-Service (PaaS) is readily available as open source (e.g., the closed code behind Amazon Web Services or Rackspace Cloud Hosting). And, while code that implements IoT applications and IoT-centric Software-as-a-Service (SaaS) solutions leverages OSS, there is no impetus for that code to be open source itself. A good analogy lies in Android. While the Android platform is derived from hundreds of open source components and is itself mostly open, and while application development tools and support libraries are published as open source, the majority of the applications distributed through Google Play are closed and proprietary.

END-USER SOFTWARE

End-user IoT software supports monitoring, control, and configuration of IoT devices, analytics of the (vast amounts of) data generated by one or more IoT end point devices. These applications also provide domain-specific functionality that relates to the functioning of one or more IoT devices (such as medical diagnosis or soil analysis trending for crop yields). End-user IoT applications are typically manifested as web applications or mobile apps, but can really come in any form, for example, as part of Big Data analytics packages.

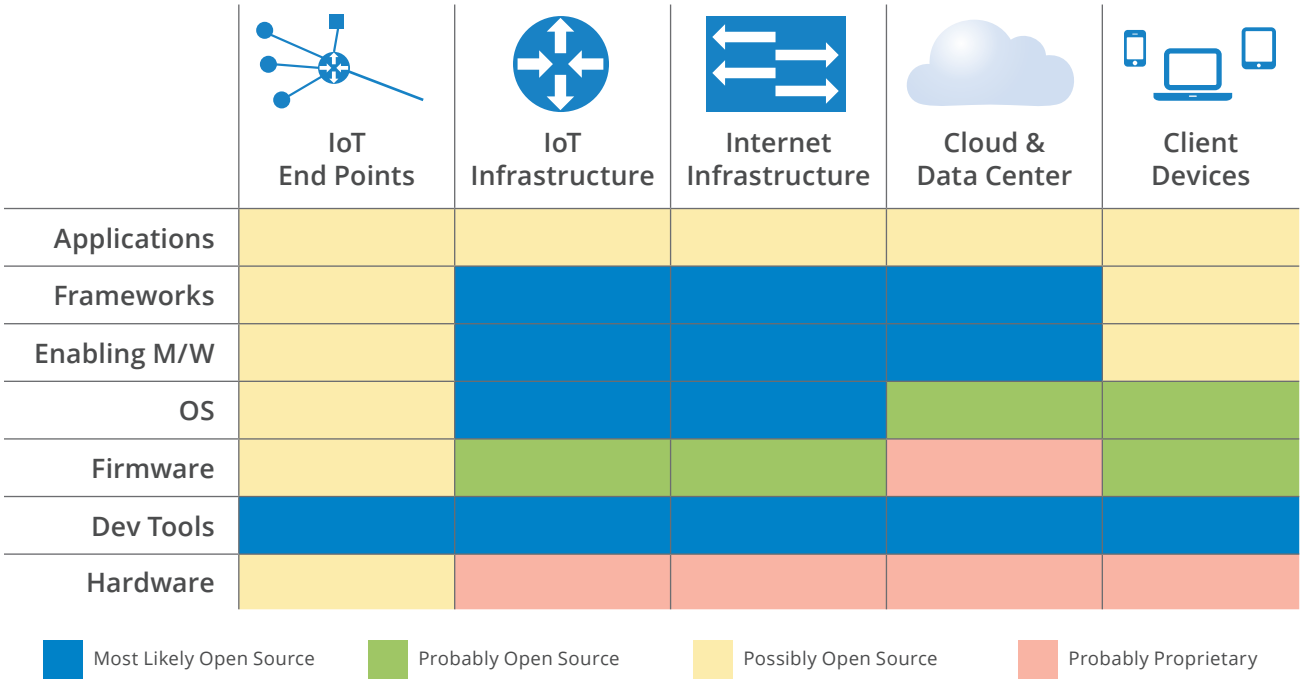

Figure 2. - Open Source in the IoT Stack

MEETING KEY IoT CHALLENGES WITH OSS: SECURITY AND PRIVACY

The history of OSS and security has been something of a roller coaster. Initially, a market accustomed to security-by-obscurity was slow to embrace community oversight to track exploits and correct the software defects that enable them. After years of debate, IT industry professionals finally came to appreciate the "many eyes" approach of OSS communities to detecting and addressing security risks (and the essential requirement to keep software up-to-date[1]). The low defect rates of OSS code were borne out by independent studies (e.g., Coverity Scans[2]). Then came the OpenSSL Heartbleed bug[3], and the pendulum began to swing back, with IT end-users again casting critical eyes upon the security of OSS code, even as community developers acted quickly to remedy the vulnerability.

Heartbleed, and more recently Shellshock, provide a wake-up call to developers and enterprise security professionals. It's now become clearer than ever that solutions are needed to automate the discovery and tracking of open source components in use within applications, along with identifying security vulnerabilities reported against specific open source components. Also important is the ability to assess which specific applications use components with known vulnerabilities.

Privacy-wise, OSS has stepped up to enable protection of individuals' data by implementing strong encryption for the masses (PGP, etc.) and by supplying the building blocks for mobile security and data protection (whether or not they are currently employed to great effect).

The Internet of Things presents its own set of security and privacy challenges:

- Myriad device types built with greatly varying degrees of security expertise
- A rich mix of public and private data (e.g., publicly available environmental sensor output vs. private health telemetry, or public aggregate data built from sensitive individual sources)
- Potential to disrupt operation of industrial and energy systems, and of life-critical connected devices

None of the above challenges is insurmountable, but there are no magic bullets either. The most ubiquitous "things" on the Internet today are mobile phones and tablets, which stand out as a morass of security problems. Mobile OEMs, system software developers, network operators, application software vendors, IT departments, and end-users find themselves playing security "whack-a-mole," despite the efforts of the global developer communities around Android and other platforms (open and proprietary).

While surely key in innovating solutions to IoT security challenges, open source is just one factor in any comprehensive IoT security and privacy paradigm. Equally important are best development practices and logistical tools to augment and enforce those practices.

CONCLUSION

That open source software will help drive the IoT build-out is obvious, but dominance in IoT technologies is not a foregone conclusion. Open source does dominate large swaths of intelligent device software, networking and network infrastructure code, and cloud platform software. For that strong position to translate into IoT dominance, developer communities will still have to cross key gaps and implement technologies essential to the Internet of Things. The good news is that with more than two million OSS projects launched to date and tens of millions of active OSS developers, meeting the unique and emerging needs of the IoT will be all in a day's work for OSS and the communities that create and comprise it.

1. See also http://www.blackducksoftware.com/solutions/open-source-security
2. *Coverity Scan Report Finds OSS Quality Outpaces Proprietary Code*
3. See http://osdelivers.blackducksoftware.com/2014/04/09/the-heartbleed-bug-what-you-need-to-know-now/

ABOUT BLACK DUCK SOFTWARE

Black Duck provides the world's only end-to-end OSS Logistics solution, enabling enterprises of every size to optimize the opportunities and solve the logistical challenges that come with open source adoption and management. As part of the greater open source community, Black Duck connects developers to comprehensive OSS resources through The Black Duck Open Hub (formerly Ohloh), and to the latest commentary from industry experts through the Open Source Delivers blog. Black Duck also hosts the Open Source Think Tank, an international event where thought leaders collaborate on the future of open source. Black Duck is headquartered near Boston and has offices in San Mateo, London, Paris, Frankfurt, Hong Kong, Tokyo, Seoul, and Beijing. For more information about how to leverage open source to deliver faster innovation, greater creativity, and improved efficiency, visit www.blackducksoftware.com and follow the company at @black_duck_sw.

CONTACT

To learn more, please contact: sales@blackducksoftware.com or 1.781.891.5100
Additional information is available at: www.blackducksoftware.com

WP-OS_IoT-UL-1014