Contribution ID: **170** Type: **not specified**

Over the Air (OTA) Updates: State of the Union? Democratize?

Monday 9 September 2019 15:30 (30 minutes)

IoT applications, be they Autonomous Cars [1] or Health Care or Smart Home or Factory Automation, the IoT devices (sensors and actuators), gateways, and cloud/datacenter endpoints need software and/or firmware updates, to fix security issues, patch bugs, and/or release new features. IoT with its numerous remote devices and gateways presents a large attack surface, making the application of security patches as they become available especially important. Let us review key OTA Update requirements, available open source solutions, and how to ease adoption through the introduction of a standard API, one that abstracts the complexities and trade-offs. The underlying implementation would be selected based on the application needs and where in IoT architectural stack a given node lies (device/edge/cloud).

Most of us are familiar with OTA in the context of our mobile phones. A large proportion of Tesla's success and customer confidence stems from its OTA update support [2], for example a braking distance issue fix, accepting only signed updates, and rolling out new features.

What are key OTA requirements [3]?
1) Ability to upgrade the bootloader, kernel, root filesystem, firmware, applications, device specific data.
2) Robust - Never "brick"the device.
3) Atomic - success or failure, nothing partial with undefined behavior.
4) Automated –not requiring human interaction during the process
5) Auditable –logs –what got updated
6) Preserve user data (customizations etc)
7) Signed, accepting updates only from trusted sources.
8) Secure communication channel.

Note: We shall drop the bootloader in item 1 because it is a transient power-on, process is rarely a source of runtime bugs.

What are OTA implementation considerations [4]?
Inline or Shadow Partition?
OTA is not easy and there are many implementation options with their respective trade-offs. Should the update happen in-place or use a shadow partition to copy over? What size should the partition be? The shadow partition approach is certainly safer just in case there is a power glitch at either the recipient or server node, or a network glitch or some other error condition that could corrupt an in-place update. Roll-back in case of a corrupted update is easy with shadow partitions because the original boot image is intact.
Block-based or file update?
The former is a complete image, easy to verify with a version number and hash signature, making it simpler to manage across a fleet of devices. The latter is more concise but should issues arise in the application of the patch, the system could become inoperable.
Trusted Source?
Can the update payload be trusted? Is it signed by a trusted entity? This requires the nodes have the public key and certificate of the trusted entity.
Where can updates be obtained?
Perhaps a vendor specific site. Possibly even a public site if open source.
Update Push or Pull? Frequency?
Should nodes poll for updates or should a management application push updates to nodes?

Secure transmission?
Is the transmission channel secured using TLS/HTTPS or over a VPN?

What open source projects address OTA?
The projects below vary in their robustness, network bandwidth needs, and the types of hardware they support.

OSTree [6,7]
Provides a git like approach to version control for Linux operating systems that does an atomic complete filesystem update. The userspace solution can operate either standalone or be layered with a package manager for a hybrid solution.

Balena.io [8]
balenaOS Yocto Linux based host OS that comes packaged with balenaEngine, a lightweight docker-compatible container engine. A device supervisor runs in its own container and allows pulling new code even if the application code crashes.

SWUpdate [9,10]
SWUpdate is a Linux Update agent with the goal to provide an efficient and safe way to update an embedded system. SWUpdate supports local and remote updates, multiple update strategies and it can be well integrated in the Yocto build system by adding the meta-swupdate layer. Supports updating FPGAs and Microcontrollers.

Swupd [11]
swupd is an operating system software manager and update program that operates at a file-level to enable verifiable integrity and update efficiency.

Mender.io [12]
An open source update manager for embedded devices based on the client-server model with security and robustness.

How can we make OTA Update Easier?

Linux is the King/Queen of IoT, be it on small form factor highly resource constrained devices or on server class gateways. The OTA implementation depends upon the node, whose selection depends upon the demands of the use case. What if we could abstract away the nuances of the implementation and ease consumption, along the lines of Libvirt [13] for virtual machines that abstracts away for Cloud orchestrators machine architecture (ARM, X86) and hypervisor implementation (KVM/Xen/ESXi/HyperV/ACRN). What if we introduce "update" akin to reboot, with configuration and action sub-commands?

```
update config source <source-url>
update config key  <add|delete|update> <name> <public-key>
update config schedule <monthly|hourly|minute> <integer>
update config log <log path> // defaults to syslog
update config verify [true|false] // verifies signature publickey
update config boot-retry-limit <integer>
update config secure [true|false]  // mutual authentication [14]
update [--secure [true|false]] [--verify [true|false]] [--source <source-url>] [--now] [--noreboot]
   // values specified override config settings
   // typically reboot after update
```

Should no update implementation exist, these methods should gracefully fail reporting an error to the default log location. An update implementation when installed overwrites the default methods, which typically report "Not implemented. Consider installing X, Y or Z"

Future Enhancements:
We defer for the future supporting more secure WiFi access for IoT and OTA such as wpa3 [15]. Also in this vein is use of secure storage media such as self-encrypting-drives and read-only memory [16].

References:

1. https://www.slideshare.net/leonanavi/software-over-the-air-sota-for-automotive-grade-linux-agl
2. https://electrek.co/2017/07/17/tesla-fleet-hack-elon-musk/
3. https://mender.io/learn/whitepapers/_resources/Software%20Updates.pdf
4. https://www.embedded.com/design/operating-systems/4461019/OTA-updates-for-Embedded-Linux–part-1––Fundamentals-and-implementation
5. https://elinux.org/Secure_OTA_Update
6. https://ostree.readthedocs.io/en/latest/manual/introduction/
7. https://samthursfield.wordpress.com/2014/01/08/os-level-version-control/
8. https://www.balena.io/what-is-balena/
9. https://github.com/sbabic/swupdate

10. http://events17.linuxfoundation.org/sites/events/files/slides/ELC2017_SWUpdate.pdf
11. https://clearlinux.org/documentation/clear-linux/concepts/swupd-about
12. https://mender.io
13. https://libvirt.org
14. https://searchsecurity.techtarget.com/definition/mutual-authentication
15. https://www.linux.com/news/wpa3-how-and-why-wi-fi-standard-matters
16. https://openiotelcna2017.sched.com/event/9J5i/surviving-in-the-wilderness-integrity-protection-and-system-update-patrick-ohly-intel-gmbh

I agree to abide by the anti-harassment policy

Yes

Primary author: Dr BHANDARU, Malini (VMware)

Presenter: Dr BHANDARU, Malini (VMware)

Session Classification: You, Me, and IoT MC