Software Updates for Connected Devices

Key Considerations

Drew Moseley
Solutions Architect
Mender.io

Session overview

1. Survey: state of updating embedded software today
   - 30+ interviews
2. Environment and criteria for embedded updater
3. Solution strategies for updating embedded devices

About me

Drew Moseley

- 10 years in Embedded Linux/Yocto development.
- Longer than that in general Embedded Software.
- Project Lead and Solutions Architect.

drew.moseley@mender.io

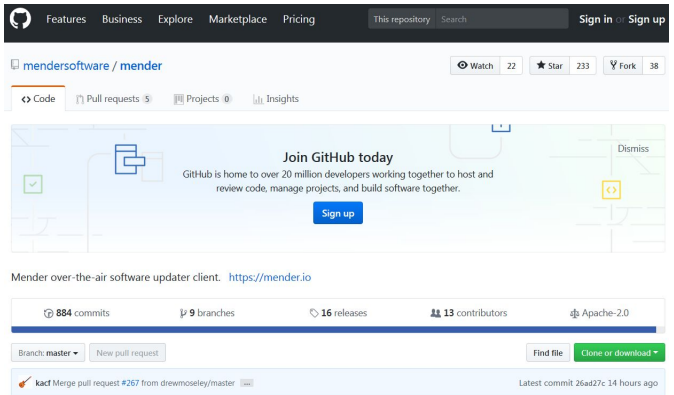
https://twitter.com/drewmoseley

https://www.linkedin.com/in/drewmoseley/

https://twitter.com/mender_io

Mender.io

- Over-the-air updater for Embedded Linux
- Open source (Apache License, v2)
- Dual A/B rootfs layout (client)
- Remote deployment management (server)
- Under active development



Connected devices must be remotely updatable

- There *will* be **bugs, vulnerabilities**
  - 1-25 per 1000 lines of code*
- … and new **features**
- … after device is **deployed to the field**

Fiat Chrysler recalls 1.4 million cars after Jeep hack

24 July 2015 | Technology

Critical- and high-severity security bugs in Linux

Source: Ars Technica

**Source: Steve McConnell, Code Complete*

We need **robust** and **secure** OTA updates

- Power loss during update
  - Atomic?
  - Automated rollback?
- Secure communication
- Signed updates
- Homegrown seems easy
  - Is it really? (hint: no)

Tesla hacked by security researchers in September 2016

"Cryptographic validation of firmware updates is something we've wanted to do for a while[…]" - Tesla's CTO JB Straubel

Vulnerability in Deutsche Telekom's updater exploited

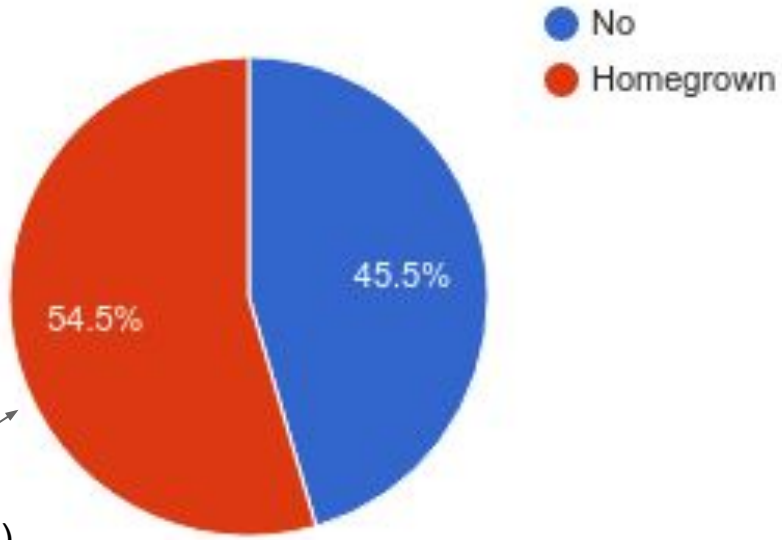
30 New Mirai Worm Knocks 900K Germans Offline

NOV 16

More than 900,000 customers of German ISP **Deutsche Telekom** (DT) were knocked offline this week after their Internet routers got infected by a new variant of a computer worm known as **Mirai.** The malware wriggled inside the routers via a newly discovered vulnerability in a feature that allows ISPs to remotely upgrade the firmware on the devices. But the new Mirai malware turns that feature off once it infests a device, complicating DT's cleanup and restoration efforts.

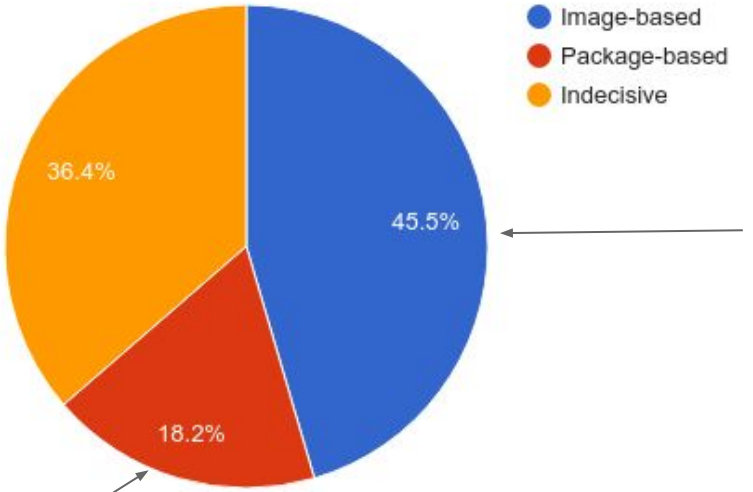
https://krebsonsecurity.com/2016/11/new-mirai-worm-knocks-900k-germans-offline/

Do you deploy updates today? How?



Includes one-by-one (standalone)

Image-based or package-based deployment?



- “Atomic”
- “Consistent”

- “Fast installation”
- “Easy to develop”
- “Uses less bandwidth”

Development time and frequency of use

- Q: How long did you spend for **initial development** for **homegrown** updater?
  - A: 3-6 months
  - Maintenance time additional

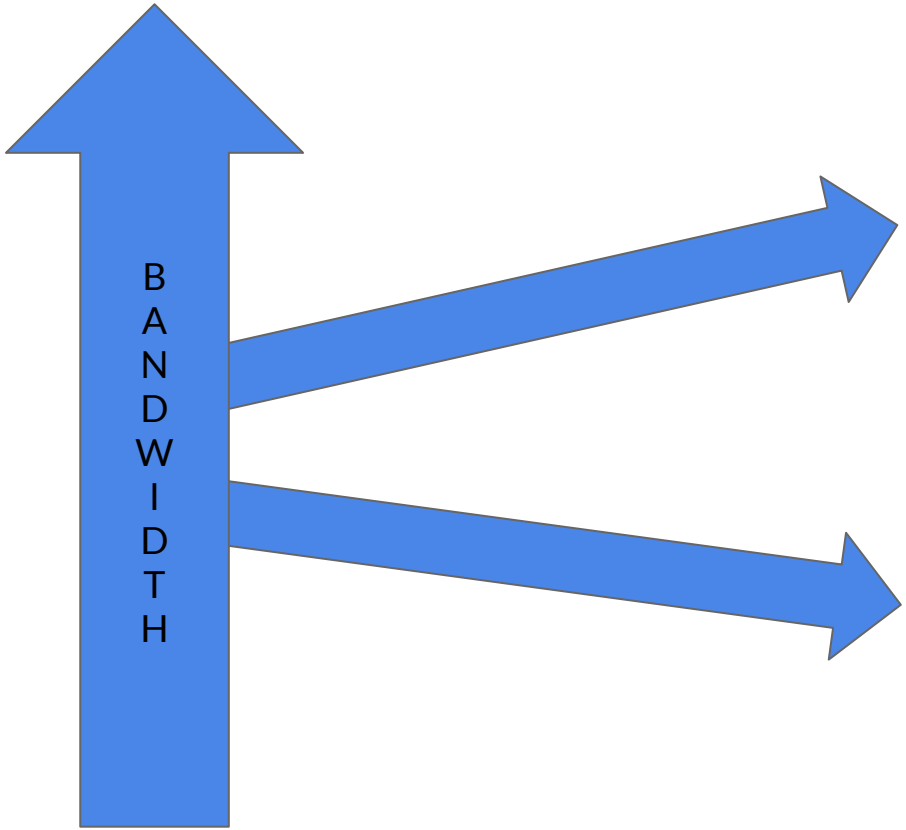
- Q: How **frequently** do you deploy remote updates?
  - A: 6 times / year

- Bottom line: most systems need improvements
  - Interest in OTA has picked up
  - Internet of Things “IoT” is the biggest driver

The embedded environment

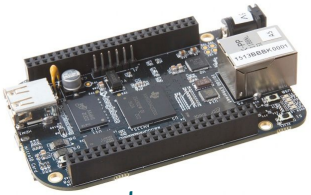
- Remote
  - Expensive to reach physically
- Long expected lifetime
  - 5 - 10 years
- Unreliable power
  - Battery
  - Suddenly unplugged
- Unreliable network
  - Intermittent connectivity
  - Low bandwidth
  - Insecure

Network requirements are *different than for smartphone*



$700

- Expensive
- High data speed
- 3G, 4G, 5G, wifi



$30

- Low cost
  - Hardware
  - Data transfer
- Small size
- Low data speed
- High connectivity

Key criteria for embedded updates

1. Robust and secure
2. Integrates with existing environments
3. Easy to get started
4. Bandwidth consumption
5. Downtime during update

1. Robust and secure

Drivers:

- Power or network loss any time
- Hostile deployment environment

Requirements:

- **Atomic** installation
- **Consistent** deployments across devices
- Sanity check **after** update
- Ensure **authenticity** of update

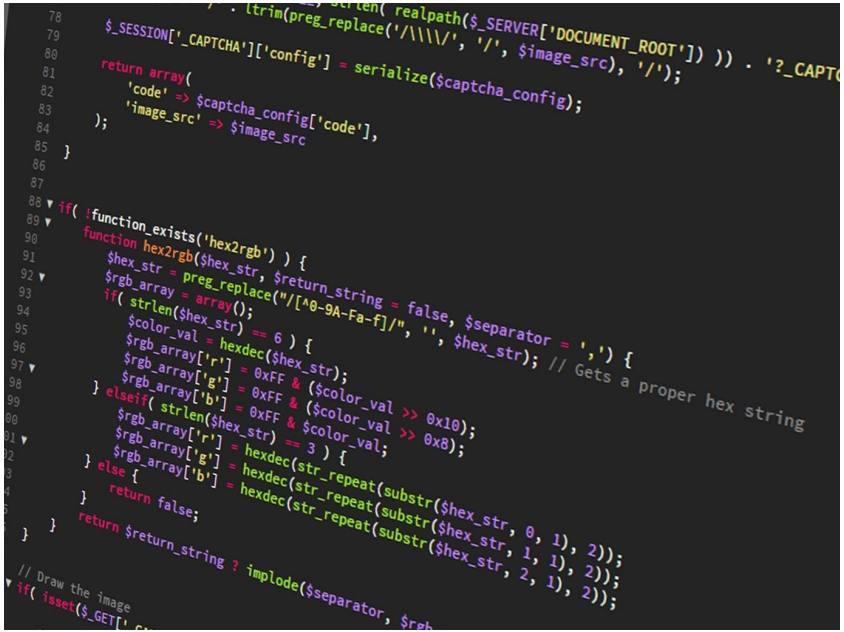
2. Integrates with existing environments

Drivers:

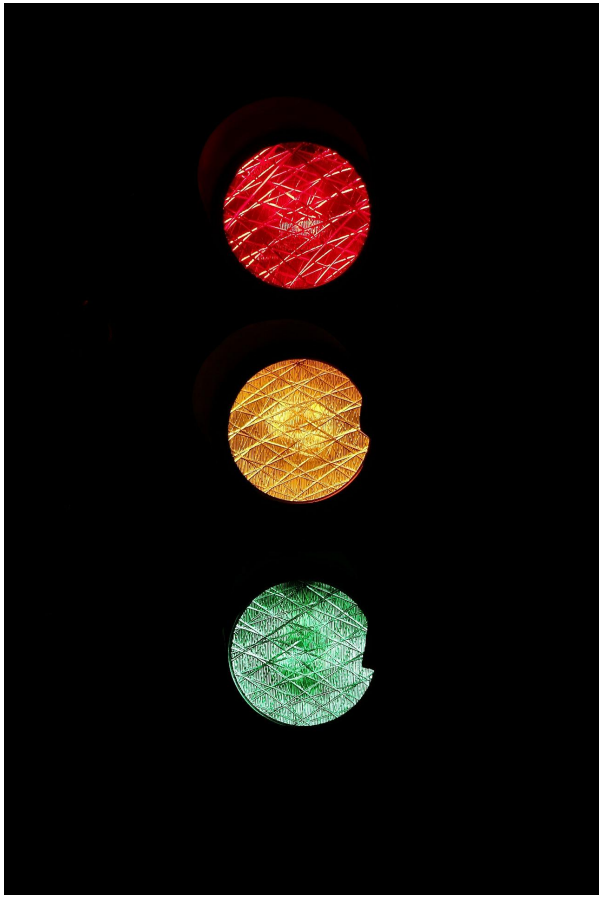
- Add OTA capability to existing projects
- Device specific use cases
- Overcome developer resistance

Requirements:

- Easy to integrate
- Standalone and managed mode
- Extensible
  - Plugins for custom actions
  - Custom installers
  - Multiple architectures and Operating Systems
  - Users & System designers can control the workflow



3. Easy to get started



Drivers:

- Quick to get started
- Overcome developer resistance

Requirements:

- Reference implementation
- Test reports
- Continuous Integration (publicly available?)
- Case studies
- Good documentation

4 & 5. Bandwidth & downtime during update

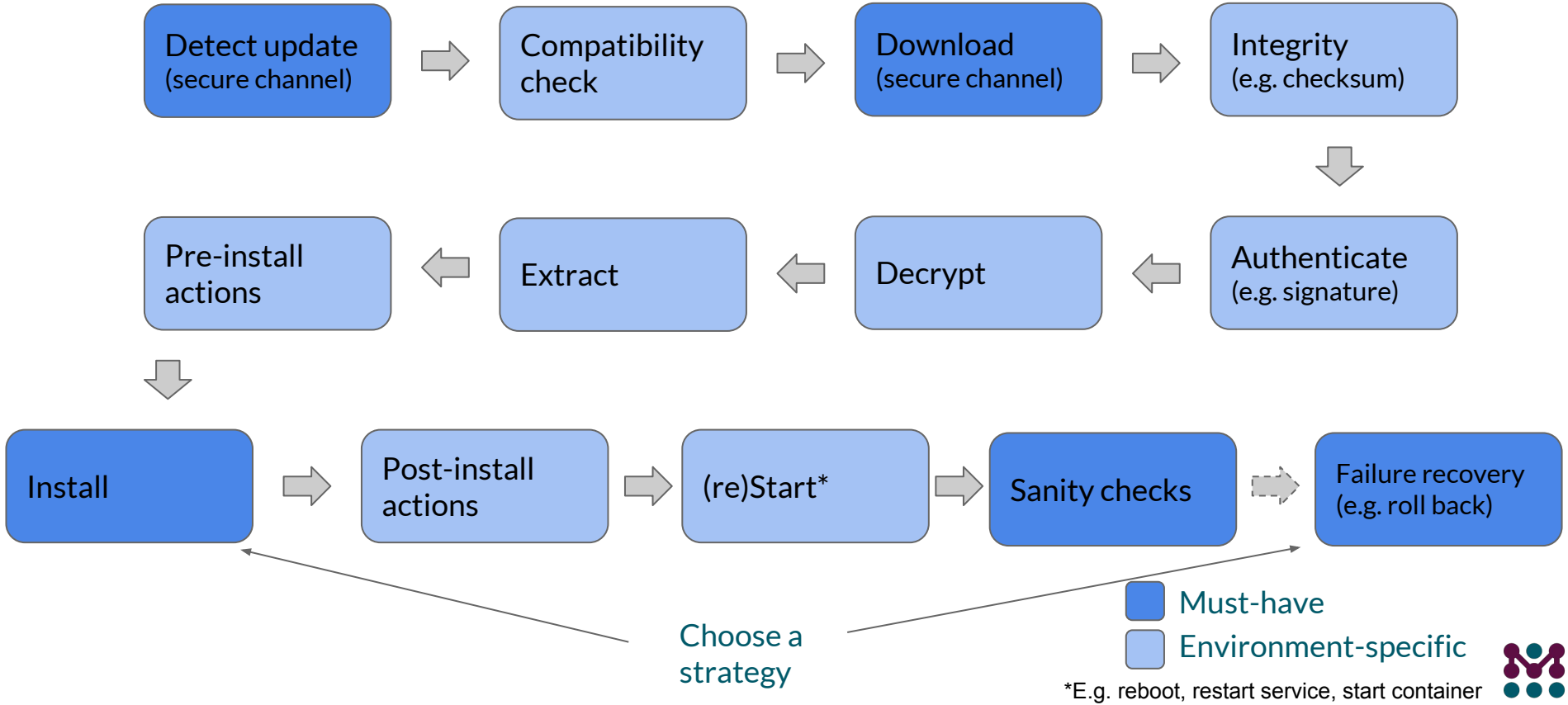
Drivers:

- Network expense
- User frustration

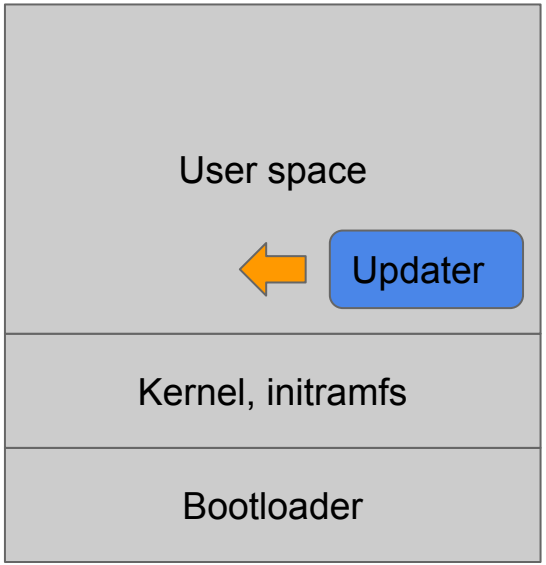
Requirements:

- Generally lower is better
- Customizable polling interval
- Maintenance windows
- Low CPU overhead

Generic embedded updater workflow



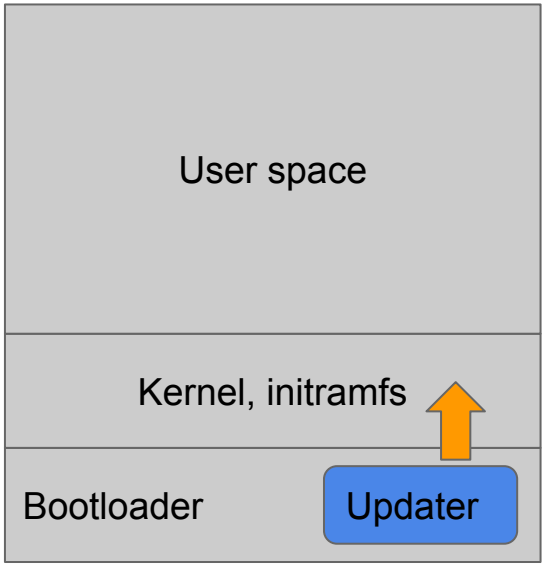
Installer Strategies (1/4)



In-place Installation: Updater deploys to a running environment

1. Robust and secure - poor
   - Atomicity: difficult or impossible
   - Consistency: difficult
2. Integrates with existing environments - good
   - Packages provided by distribution
   - Scripting for building packages
3. Easy to get started - good
   - Tightly integrated with distribution
4. Bandwidth consumption - good
   - Transfers only updated files
5. Downtime - good

Installer strategies (2/4)

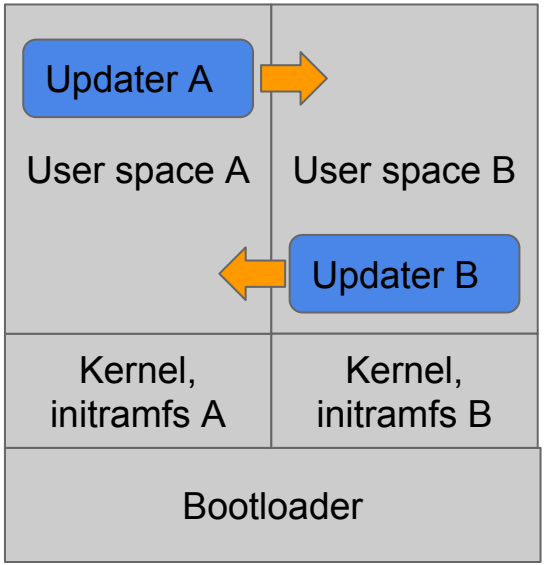


Asymmetric maintenance mode

1. Robust and secure - poor
   - Atomicity requires extra work
   - Consistency: good on successful update
2. Integrates with existing environments - fair
   - No runtime modification
   - Requires boot loader modifications
3. Easy to get started - fair
   - Requires extensive boot loader modifications
4. Bandwidth consumption* - poor
   - Full image
   - Two transfers in the case of roll-back
5. Downtime - poor
   - System is down during update.
   - Two updates in the case of roll-back

*Can mitigate: compressed/delta

Installer strategies 3/4

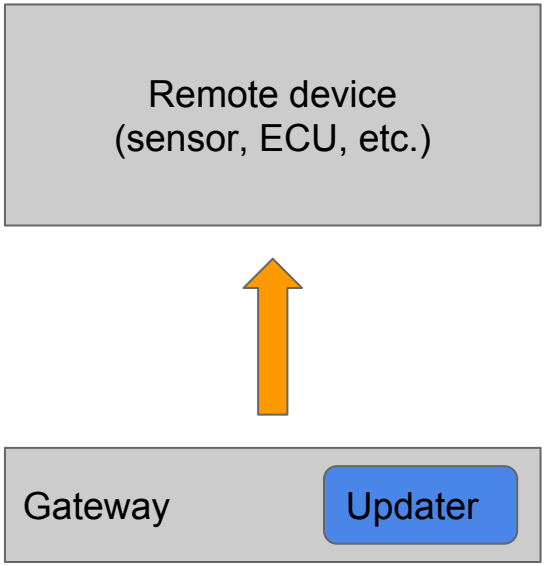


Symmetric dual A/B rootfs

1. Robust and secure - good
   - Fully atomic and consistent
2. Integrates with existing environments - fair
   - OS, kernel, apps unchanged
   - 2x rootfs storage
   - Runtime client needed
   - Minor boot loader modifications
3. Easy to get started - good
   - Application/system code unchanged
4. Bandwidth consumption* - poor
   - Full image
   - Only one transfer even with roll-back
5. Downtime - good
   - Asynchronous Install
   - 1 reboot downtime (or 2 with roll-back)

*Can mitigate: compressed/delta

Installer strategies 4/4



Gateway device acts as proxy

- Different scenario
  - Smaller devices (no client)
  - Complements other strategies
- Local installations (ie not internet based).
- Gateway must handle robustness and security

Comparison of installer strategies

	1. In-place	2. Boot to maintenance mode	3. Dual A/B rootfs
Atomic/Consistent			
Workflow integration			
Bandwidth			
Downtime			

Deployment - Managed Mode

- Centrally manage remote devices
  - Reporting
  - Scalability
  - Integration with device management infrastructure

- Operator connects to and controls clients
  - Device groupings
  - Campaign management

Ensure your devices can be updated remotely

- Rest assured you can fix that undiscovered bug!
- Consider your update strategy early in your design cycle
- Choose installer strategy that fits your environment the best
- Use third party or open source tools where possible
  - Home grown is more difficult than it seems

Thank You!

Q&A

@drewmoseley

https://mender.io

drew.moseley@mender.io