Memfault

A Consolidated Approach to Managing IoT Devices:

Using an Observability Platform to Get Better Connected Devices to Market Faster

White Paper | memfault.com

Using an Observability Platform to Get Better Connected Devices to Market Faster

IoT businesses across every industry—from consumer electronics to industrial automation—must navigate device development, production, maintenance, operations, and repair within an increasingly connected and dynamic environment. And the complexity of the IoT ecosystem will only increase as the number of devices continues to grow at an increasingly swift rate.

There's no question that the rise in connectivity has delivered significant utility through new and innovative use cases. From smart homes to industrial facilities, connected devices have become commonplace with a range of functionality that covers the convenient to the life-saving, from toothbrushes to patient monitoring and wildlife conservation. And with the rise in IoT device use cases comes rising expectations that these devices work, and *work well.*

Because of its growing usefulness and reach, IoT connectivity introduces complications for which traditional hardware/firmware processes have no clear remedies. The consequences of shipping a product blindly without a clear plan for updating and patching it quickly have greater impact not just technologically but also on a company's bottom line.

Software teams may have access to a suite of tools that allow for agile development, troubleshooting, and updating, but hardware teams do not. They're often juggling a patchwork of solutions or spending valuable engineering hours and dollars developing custom in-house solutions for managing connected devices. Most solutions are siloed—addressing just one part of the product life cycle but lacking the capability of end-to-end device observability that unifies the monitoring, updating, and debugging of devices.

Without an effective, consolidated approach to managing IoT devices, businesses are forced to accept a number of costly realities:

- **Releasing products slower than necessary,**
- **Shipping bugs with no way to receive alerts of them and no way to fix the issues**
- **Being bound to a passive bug notification system that turns end users into beta testers, and**
- **Relying on a hope-and-see approach to ongoing device operations with no active improvement plan or schedule.**

Proactive management of your infrastructure, with end-to-end observability for your connected devices, is a must.

ABI Research estimates that there are 8.6 billion IoT connections.

By 2026, they expect the number to nearly triple to 23.6 billion.

Move Away from a Disjointed, Outdated Process

Historically, developers could write static firmware for specific device use cases or commoditized products and, once the product was released, have no further interaction or engagement with the product. With little visibility into device health, updates were released based on engineering's schedule or product roadmap, or when a bug had been reported by a critical mass of end users.

Limit Disruption

Greater connectivity means a rise in device performance expectations. It also requires a shift in approach. Device companies look to limit disruption to end users. Engineers are under pressure to deliver updates behind-the-scenes—at a time, and in a manner that's nearly imperceptible to the end user. Today's connected devices require ongoing updates to help fix bugs, add new features, and patch known issues. They also require a process for updating that is agile and proactive to ensure ongoing device security and prevent users' data and privacy from being compromised. Without continuous visibility into fleet health and a remote debugging solution that preempts potential issues, companies cannot deliver these updates successfully. And as devices are increasingly part of the subscription model ecosystem that drives software-as-a-service (SaaS) companies, regularly ensuring they work properly reduces the potential of lost revenue that's a result of device malfunction.

Consider each of these questions about **your strategy for delivering device updates:**

- How will the device be updated?
- When should updates be pushed?
- How can updates be reverted or rolled back?
- How are issues in deployments identified?
- Can updates be rolled out on a limited number of devices?
- How are security issues identified and fixed before any customer information is compromised?

Greater connectivity means a rise in device performance expectations.
It also requires a shift in approach.

Eliminate Waste

The truth is, everyone ships bugs. Everyone.

After detecting a bug, the process for debugging them is often inefficient and expensive. To fix them, traditionally firmware engineers have needed the physical device returned or they would have to blindly reproduce and solve the issue with little information.

Often, with a debugger in hand, teams would spend valuable staff time reproducing to source it. **Most connected device developers estimate that 40–50% of their time is spent on diagnosing and debugging,** and that's only for the issues you actually hear about! But when you acknowledge that the tools hardware teams have access to were not purpose-built for IoT or firmware, this statistic is not that surprising.

A system that presupposes regular and ongoing re-engineering is a system that experiences significant waste, in shipping costs, in engineering time, in the potential loss of repeat customers. Developers are forced to spend their time responding to issues, rather than actively innovating and improving.

Consider each of these questions about **your strategy for debugging devices:**

- Will the development team get instant notifications when a bug occurs or will they have to rely on RMAs?
- Do you have access to automatic data extraction and aggregation capabilities?
- Are you able to discern whether a bug is new, or whether it'll be resurfaced after a firmware update?
- Can you prioritize issue resolution based on the bugs affecting the most devices?
- Do you have access to the same hard data you would with your physical debugger?

A system that presupposes regular and ongoing re-engineering is a system that experiences significant waste, in shipping costs, in engineering time, **in the potential loss of repeat customers.**

Establish Observability

Connectivity, performance, and/or battery-life regressions are occurring, but unless the issues are severe, and users are highly motivated to report them, developers often aren't aware of potential fleet-impacting bugs. Related patterns can be difficult to replicate; unforeseen impacts of firmware updates across versions can be hard to trace and source. Lacking insight into overall fleet health, developers often are limited to acting only when notified of critical issues, rather than continuously monitoring for smaller issues and addressing them before they're discovered by end users.

Consider each of these questions about **your strategy for monitoring devices:**

- Was the latest firmware release an improvement or regression?
- What does the device's historical timeline indicate?
- When did the issue begin, and are other devices experiencing it?
- How many devices are operating with the most recent version?
- Is there visibility into the key metrics most important to fleet health?
- Did we fix the issues we expected to fix with a release?

Once connected devices are in production, **monitoring them is critical, but not easy.**

Meet Memfault

Reduce risk. Ship products faster. Resolve issues proactively by upgrading your connected device infrastructure with Memfault.

Founded by embedded engineers frustrated with the lack of agility and flexibility of hardware tools, Memfault is the first cloud platform designed to manage the full device lifecycle. Memfault combines the three key functions of device development and operations into a consolidated observability platform. Through a single admin console, Memfault allows companies to:

1. Easily monitor both device and fleet-level metrics,
2. Push new OTA updates to users,
3. Quickly debug any issue that arises.

Memfault's integration of updating, monitoring & debugging into a single platform leverages data across capabilities features that allows for better products faster, that improve over time, without compromising stability reliability.

Key Benefits of using Memfault

1. Better visibility into fleetwide device health, via continuous monitoring of device metrics;
2. Faster time to market, through iterative developments and updates that can be pushed instantaneously even after shipping—no product development freeze required;
3. Improved products, through responsive development built on fleet-wide feedback and operations;
4. End-user convenience, provided by automatic updates and controlled releases with specific cohorts; and
5. Swift innovation, which empowers engineers to spend time getting the product out and pushing new features, instead of debugging and troubleshooting.

Memfault OTA Firmware Updates

- Offer incremental rollouts, so developers can start with a small number of devices and ramp up over time to limit impact of any new issues.
- Insulate users from repeated, successive bugfix releases in a row.
- Enable A/B updates, limiting user impact to a simple reboot prompt.
- Pair updates with monitoring for swift problem identification and the capability to pause or abort updates as necessary.

Memfault Remote Debugging

- Avoid product freezes in favor of agile development flows and continuously improved algorithms.
- Ship minimum viable product (MVP) through Day-0 updates and iterate after product shifts.
- Root cause issues with hard data and automatically catch and triage bugs.

Memfault IoT Device Monitoring

- Accelerate collection and analysis of crashes, assertions, and other errors occurring in the field.
- Measure the health of an entire fleet of devices across all dimensions (release version, groups of devices, firmware version, etc.)
- Monitor millions of devices.
- Easily monitor fleet health at scale.

A New Virtuous Cycle

We've already reached the point of connected devices outnumbering non-connected devices. As connected devices impact ever greater areas of consumer and industrial life, businesses will be under increasing pressure to ensure that the end user experience is simple and secure.

Centralizing the monitoring, updating, and debugging of IoT devices empowers developers to ensure that they're shipping the best product they can, as fast as they can, with confidence that errors and issues can be solved swiftly and seamlessly. Memfault makes that happen.

Try Memfault

Learn more about Memfault by visiting **memfault.com** or get started with Memfault today by signing up free here: **memfault.com/register**

Memfault Case Studies

01

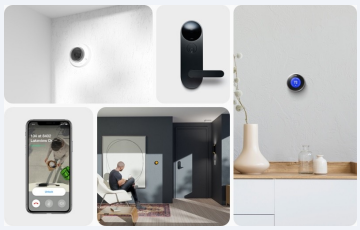
Mobile Motion Tech
with Diamond Kinetics

02

Wellness Wearables
with Silvertree

03

Building Operating System
with Latch



Memfault