**Buggy software should be fixed first, delivered later**

1. Establish a standardized process.

It's important to establish a standardized process to address bugs.

1. Make plans to quickly fix defects.

Another approach to fixing bugs in production is to work backward from bug identification, to how the team could swiftly address such defects.

1. Practice time management.

Bug fixes are faster and less disruptive to production when the team has a well-planned approach.

1. Implement benchmarks.

Software teams should use benchmarks to estimate how many bugs the team can fix in a month. For example, in the U.S., an average programmer can fix between nine and 10 bugs in a month, Crippen said. An experienced programmer can fix up to 20 bugs in that time. With these averages in mind, IT leaders can estimate how many bugs the team can tackle.

1. Prioritize test code.

A development team can prioritize the code it uses for testing at the same level as production code. The result is that fewer bugs will slip through to live environments.

1. Perform chaos engineering.

Software testing verifies the code does what it's supposed to. However, such QA can miss bugs caused at the level of the systems where the code runs. [Chaos engineering](https://www.techtarget.com/searchitoperations/definition/chaos-engineering) hits software -- usually live in production or in a realistic staging environment -- with unpredictable disruptions.

1. Move fast and break things.

A company might take a relaxed attitude about releasing production code with bugs if business growth and popularity depend on pushing out new functionality quickly.

1. Adopt a mission-critical mentality.

Companies should avoid the former section's approach when building [mission-critical](https://www.techtarget.com/searchitoperations/definition/mission-critical-computing) software like avionics, autonomous cars or medical equipment. "You're going to have to adhere to extremely rigorous processes if peoples' lives or expensive equipment are at stake," Wade-Stein said. A mission-critical mindset helps developers build software that enhances the product's brand reputation. However, the extra precautions are expensive.

1. Mature the product.

In early stages of the product lifecycle, teams can lose valuable time to market by focusing on perfect releases. The attitude toward fixing bugs in production can shift, depending on where you are in the product's lifecycle, which provides goal management software.