[Extended Abstract] Software doesn't make a noise when it breaks

Piyush Verma
piyush@piyushverma.net
Last9, India

This talk is a tale of few such failures that went right under our noses and what we did to prevent those. The failures covered in the talk range from Heterogenous systems, unordered events, missing correlations, and human errors

I will take some curious failures that we have dealt with in the past decade of my work with Infrastructure systems and techniques and software we had to build to:
- Isolate
- Limit the spread
- Prevent from happening again

An un-replicated consul configuration results in data loss 25 hours before a countrywide launch. Took a staggering 5 engineers and 20 hours to find one single line of change.

A failed distributed lock in etcD. Forcing us to re-write the whole storage on Consul and hours of migration. Only to find out later that it was a clock Issue.

The above Isolation and immediate fixes were painfully long, yet doable. The real ambition was to prevent similar such Incidents from repeating. I will share samples of some of our RCAs and what was missing with each one of those versions. And what the resultant RCA looks like. This section does touch briefly upon blameless RCA but real point of focus is action-ability of an RCA.

In this section, I will showcase some of the in-house frameworks and technologies (easy to replicate) that were built to turn the prevention/alert section of RCAs into lines of code rather than lines of blurb of text. The goal of this section is to showcase and advocate the need to build/adopt tool-chains which promise early-detection and not just faster-resolution.