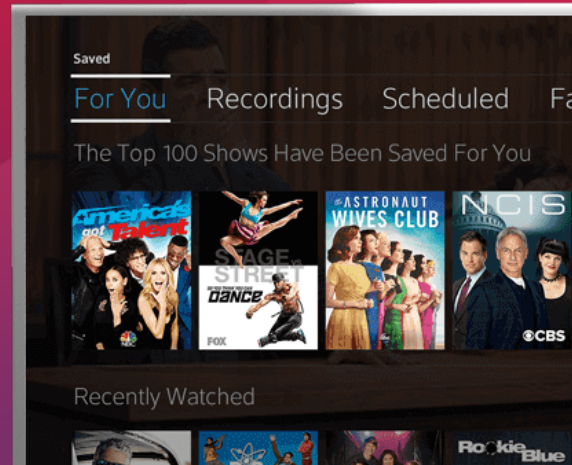


# How Comcast Ensures Application Reliability to Serve Over 10M Video Customers

Using OverOps on its flagship X1 XFINITY platform helps Comcast stay on top of every new error



XFINITY's X1 is the flagship application for Comcast, providing users with a different television experience. The application offers an interactive platform combining universal search results from live TV, Comcast's On Demand programming, and DVR recordings, in addition to personalized recommendations and apps. It runs on 23 million devices, across dozens of different data centers.

153k+  
Employees

23M  
Devices

\$75B  
Revenue

180.5B  
Total Assets

Production Monitoring Ecosystem:

OverOps

splunk>

ZABBIX

slack

## Highlights

- OverOps helps Comcast find all of the unknown error conditions that the company didn't foresee
- Comcast uses OverOps to monitor their flagship X1 XFINITY platform
- With a weekly release cycle, OverOps helps Comcast stay on top of every new error
- OverOps is part of Comcast's automated deployment model, that helps the company instrument their application servers

## Key challenges and pain points

We offer services for over 23 million boxes, which means that an issue in production impacts a lot of users. Since we deploy a new version of our application on a weekly basis, we have to stay on top of every new error and exception that might impact the application's performance.

Our monitoring method was inconsistent. We had a log management tool with predefined queries to detect errors and exceptions, and we would spend a lot of time going through the logs trying to identify their severity level and which are worth investigating.



One person's set of go-to queries wasn't the same as the next person. Often times different people were looking at different things. And this was relevant only for a handful of the alerts. Our scale led to these alerts not being great in terms of their effectiveness, and sometimes we disregarded them since they were noisy. This was a tedious process, that involved manual effort from our team.

Since there are millions of devices that run our application, pinpointing a single error or trying to reproduce it takes up too much time and resources.

**“OverOps is used for all of the unknown error conditions that we didn't foresee”**

When issues hit production, it would impact our customers and it was up to us to try and figure out what went wrong and how to quickly fix it.

---

## How OverOps helped you solve issues

We've integrated OverOps with our automated deployment model, that helps us instrument our application servers.

We use OverOps regularly for all of the unknown error conditions that we didn't foresee. It helps us automate the process of sifting through log files, making it easier to detect issues as soon as they appear.

**“We use OverOps to monitor our flagship X1 XFINITY platform. We deploy a new version of our application on a weekly basis, so we have to stay on top of every new error.”**

In fact, we had a full day where the whole team did what we called an “exception burn-down day”,

where we basically spent an entire day fixing exceptions and log errors that were identified by OverOps. We spent a good amount of time essentially reducing the noise in our application and in a lot of cases fixing problems that had eluded us in the past.

**“With OverOps, we now have visibility into the long tail of problems that the system has, that we otherwise wouldn't have visibility into”**

Thanks to OverOps, we now have visibility into the long tail of problems that the system has, that we otherwise wouldn't have visibility into. We know as soon as an error occurs and have the ability to react fast to every issue, error or exception.

---

## How are you integrating OverOps with your daily workflow?

After installing OverOps, we almost immediately saw detailed data about our application's performance. We were able to detect where exceptions were thrown, and could display changes in the application's behavior. OverOps is especially helpful when it comes to issues that might impact our users.

We use the OverOps dashboard to see our application's behavior and look for trends along with some of the aggregated metrics. That way, we can look for highly problematic areas and quickly detect and fix any error without harming the user's experience.





### Full code and variable state to immediately reproduce any error.

No need to manually reproduce issues by searching for information in logs.

Reduce MTTI by 90%+



### <1% overhead in production

OverOps operates between the JVM and processor level enabling it to run in staging and production.



### Proactive detection of all new and Critical errors

New issues are detected and routed to the right developer vs. discovered by users. Each error receives a unique code fingerprint unique it across the app.



### No change to code or build

New issues are detected and routed to the right developer vs. discovered by users. Each error receives a unique code fingerprint unique it across the app.

---

## Supported Platforms:

JDK 1.6 and above | HotSpot, OpenJDK, IBM JVM | Java, Scala, Clojure, Groovy | Linux, OS X, Windows | Docker, Chef, Puppet, Ansible | Coming soon: .NET

## Integrations:

SLF4J, Log4j, Logback, Apache Commons Logging, Java Logger | Splunk, ELK, SumoLogic, and any other log management tool | AppDynamics, New Relic, Dynatrace | Workflow automation: Slack, HipChat, JIRA, Pagerduty | Webhooks | StatsD

---

Learn how OverOps can help you automate your deployments -  
**Schedule a demo** with an OverOps monitoring engineer

