HOW HOUSEPARTY USES TESTRTC AS AN INTEGRAL PART OF ITS WEBRTC TESTING

HOUSEPARTY:

A mobile group video chat application, where groups of up to eight friends gather to chat in virtual rooms. With over half a billion video chats conducted using WebRTC

PRODUCT:

testRTC Testing (Continuous Integration)

OBJECTIVE:

Infrastructure regression testing in varying network conditions

BENEFITS:

- Simplicity of the service
- Controlling network configurations
- Define media related success criteria
- Understand how network behavior affects the application

Houseparty selected testRTC for its WebRTC infrastructure regression testing through continuous integration.

Houseparty

Houseparty is a mobile group video chat application, where groups of up to eight friends gather to chat in virtual rooms. With over half a billion video chats conducted using WebRTC, Houseparty is massive in its scale. What makes Houseparty interesting, is that the majority of its users base are 24 old or younger audiences, spending upwards of 50 minutes a day inside the app.

Being a social platform, Houseparty has to innovate on a daily basis. This calls for frequent updates of its mobile applications and infrastructure. An update to the Houseparty backend infrastructure happens on a daily basis and the mobile apps are updated every two weeks on average.

IN SEARCH OF A REGRESSION TESTING TOOL:

The developers at Houseparty wanted to get a kind of an early warning system in place. One that would tell the team if the changes being made are breaking the service for its users. And breakage here means a reduction in media quality or the inability to work in certain network conditions. What Houseparty's developers were looking for was higher confidence in their version rollouts.

Houseparty already had stress testing capabilities in place, along with the ability to test their mobile applications. What they were missing was regression testing for the infrastructure. When a decision had to be made, Houseparty preferred to use testRTC's testing service instead of building their own testing environment, saving months of effort of experienced WebRTC developers with the understanding that the end result would be inferior in terms of its feature set and capabilities.

By selecting testRTC, Houseparty's developers were able to improve their confidence level when upgrading the service for their millions of users.

testRTC offered us the fastest and cheapest way to get the type of regression testing we needed, increasing the confidence we had when rolling out new releases of the Houseparty application.

Gustavo García,
Software Engineer at Houseparty

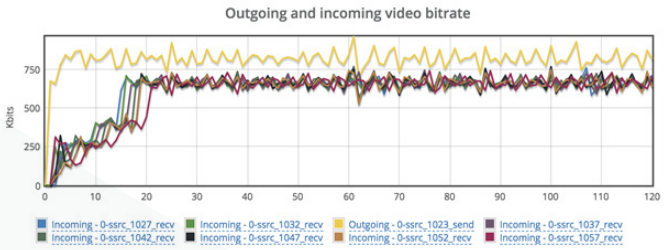
SIMPLICITY IS KEY:

One of the key reasons for selecting testRTC was the simplicity of the service. From writing tests, through selecting the machines' configuration and defining test success criteria down to integrating with the API.

The ability to pick different network configurations was really important to Houseparty. Using both the preconfigured settings as well as dynamically modifying network conditions enabled Houseparty to quickly and efficiently understand how the behavior of their application is affected.

Furthermore, by using test expectations in testRTC, a mechanism that lets developers set success and failure criteria for a test, based on metrics collected, Houseparty developers are alerted when results needs to be further analysed. This enables Houseparty's developers to spend more time on their application and less in drilling down to results, trying to understand their meaning.

When drilling down to results is needed, then the graphs displayed assist the developers in debugging the problems and resolving



Outgoing and incoming video bitrate for an 8 people room with simulcast enabled

MOBILE ONLY AND WEBRTC:

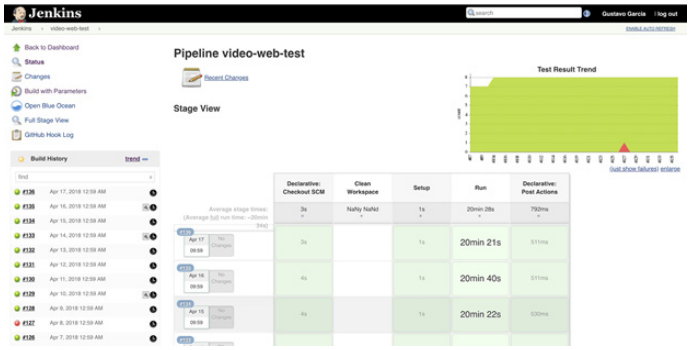
While running predominantly as a mobile application, Houseparty's video processing makes use of WebRTC. Houseparty is making a distinction between its application testing and infrastructure testing. It had in its arsenal existing tools that are being used to test its mobile clients. What it was looking for was a way to test their video infrastructure - their media servers and TURN servers - making sure they work as expected.

To that end, Houseparty is using a simple HTML page that can be used to create calls on its staging environment for the application. testRTC is then used to access that page and automate the testing process, simulating different network conditions while testing Houseparty's video infrastructure.

CONTINUOUS INTEGRATION AS A FIRST PRIORITY:

Houseparty made the decision early on to use testRTC as part of their continuous integration environment. Using testRTC's APIs, the developers at Houseparty were able to quickly integrate the testing scripts they've written in testRTC to their Jenkins automation server.

This allows Houseparty to run the testRTC regression tests every night. Integrating testRTC with Jenkins means that when tests complete, their results are reported back to Jenkins and from there they get sent to Slack, where developers get notified on potential failures.



Running testRTC tests nightly from Jenkins with integrated reporting and notifications

MOVING FORWARD WITH TESTRTC:

For Houseparty, the work is not done yet. testRTC is used on a daily basis, running a battery of tests designed to check their infrastructure. There are additional tests that are planned to be added to this test suite.

Peer-to-peer testing and direct TURN server testing will be added in the near future, increasing the coverage of regression testing done over testRTC.