Synthetic Monitoring Tests

Synthetic Monitoring Tests are a set of functional tests that target a live system in production. The focus of these tests, which are sometimes named "watchdog", "active monitoring" or "synthetic transactions", is to verify the product's health and resilience continuously.

Why Synthetic Monitoring Tests

Traditionally, software providers rely on testing through CI/CD stages in the well known testing pyramid (unit, integration, e2e) to validate that the product is healthy and without regressions. Such tests will run on the build agent or in the test/stage environment before being deployed to production and released to live user traffic. During the services' lifetime in the production environment, they are safeguarded by monitoring and alerting tools that rely on Real User Metrics/Monitoring (RUM).

However, as more organizations today provide highly-available (99.9+ SLA) products, they find that the nature of long-lived distributed applications, which typically rely on several hardware and software components, is to fail. Frequent releases (sometimes multiple times per day) of various components of the system can create further instability. This rapid rate of change to the production environment tends to make testing during CI/CD stages not hermetic and actually not representative of the end user experience and how the production system actually behaves.

For such systems, the ambition of service engineering teams is to reduce to a minimum the time it takes to fix errors, or the MTTR - Mean Time To Repair. It is a continuous effort, performed on the live/production system. Synthetic Monitors can be used to detect the following issues:

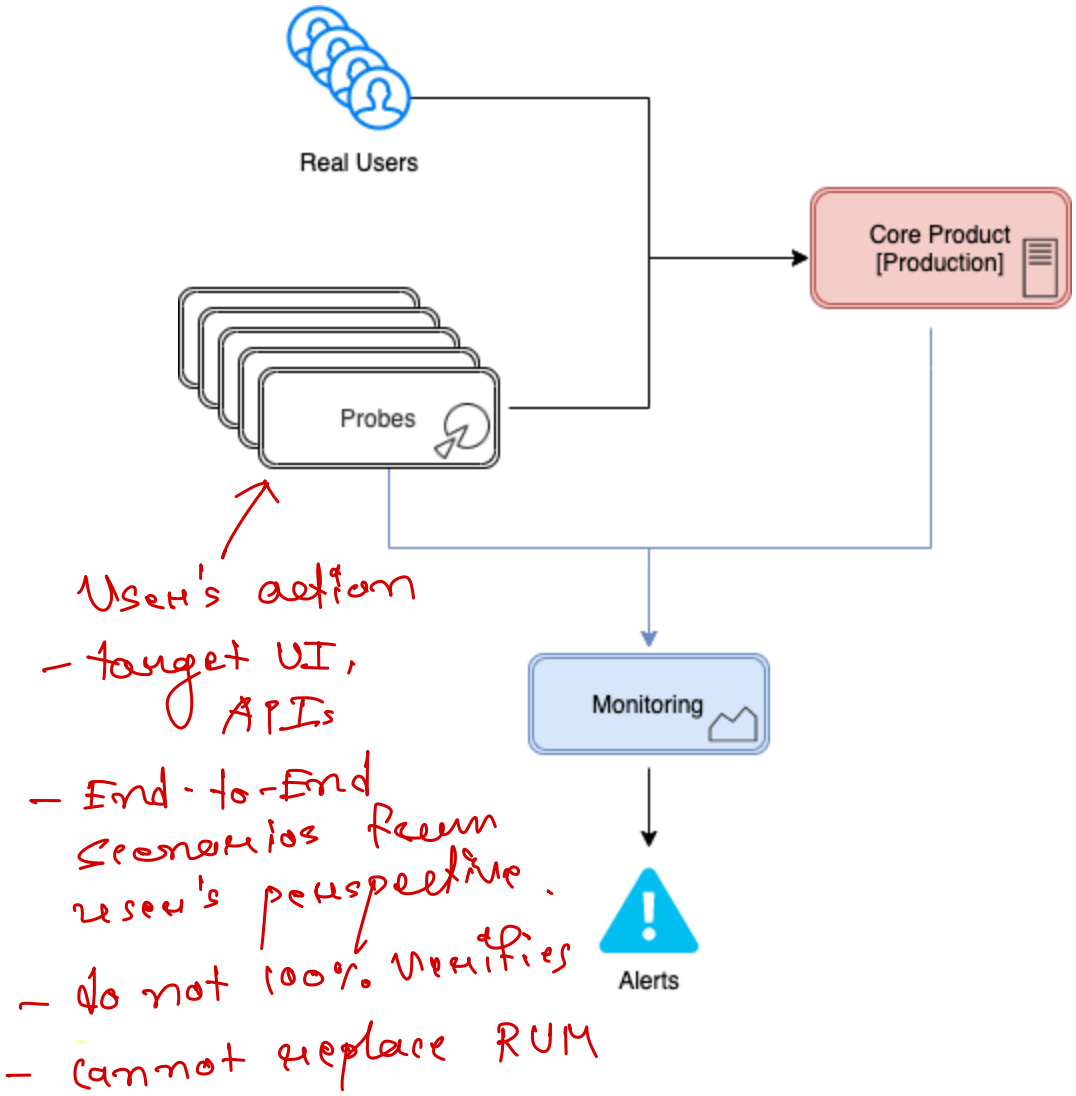
- Availability - Is the system or specific region available.
- Transactions and customer journeys - Known good requests should work, while known bad requests should error.
- Performance - How fast are actions and is that performance maintained through high loads and through version releases.
- 3rd Party components - Cloud or software components used by the system may fail.

Shift-Right Testing

Synthetic Monitoring tests are a subset of tests that run in production, sometimes named Test-in-Production or Shift-Right tests. With Shift-Left paradigms that are so popular, the approach is to perform testing as early as possible in the application development lifecycle (i.e., moved left on the project timeline). Shift right compliments and adds on top of Shift-Left. It refers to running tests late in the cycle, during deployment, release, and post-release when the product is serving production traffic. They provide modern engineering teams a broader set of tools to assure high SLAs over time.

Synthetic Monitoring Tests Design Blocks

A synthetic monitoring test is a test that uses synthetic data and real testing accounts to inject user behaviors to the system and validates their effect, usually by passively relying on existing monitoring and alerting capabilities. Components of synthetic monitoring tests include **Probes**, test code/ accounts which generates data, and **Monitoring tools** placed to validate both the system's behavior under test and the health of the probes themselves.



Probes

Probes are the source of synthetic user actions that drive testing. They target the product's front-end or publicly-facing APIs and are running on their own production environment. A Synthetic Monitoring test is, in fact, very related to black-box tests and would usually focus on end-to-end scenarios from a user's perspective. It is not uncommon for the same code for e2e or integration tests to be used to implement the probe.

Monitoring

Given that Synthetic Monitoring tests are continuously running, at intervals, in a production environment, the assertion of system behavior through analysis relies on existing monitoring pillars used in live system (Logging, Metrics, Distributed Tracing). There would usually be a finite set of tests, and key metrics that are used to build monitors and alerts to assert against the known SLO, and verify that the OKR for that system are maintained. The monitoring tools are effectively capturing both RUMs and synthetic data generated by the probes.

Applying Synthetic Monitoring Tests

Asserting the System under Test

Example

Synthetic monitoring tests are usually statistical. Test metrics are compared against some historical or running average with a time dimension *(Example: Over the last 30 days, for this time of day, the mean average response time is 250ms for AddToCart operation with a standard deviation from the mean of +/- 32ms)*. So if an observed measurement is within a deviation of the norm at any time, the services are probably healthy.

Building a Synthetic Monitoring Solution

At a high level, building synthetic monitors usually consists of the following steps:

- Determine the metric to be validated (functional result, latency, etc.)
- Build a piece of automation that measures that metric against the system, and gathers telemetry into the system's existing monitoring infrastructure.
- Set up monitoring alarms/actions/responses that detect the failure of the system to meet the desired goal of the metric.
- Run the test case automation continuously at an appropriate interval.

Monitoring the Health of Tests

Probes runtime is a production environment on its own, and the health of tests is critical. Many providers offer cloud-based systems that host such runtimes, while some organizations use existing production environments to run these tests on. In either way, a monitor-the-monitor strategy should be a first-class citizen of the production environment's alerting systems.

Synthetic Monitoring and Real User Monitoring

Synthetic monitoring does not replace the need for RUM. Probes are predictable code that verifies specific scenarios, and they do not 100% completely and truly represent how a user session is handled. On the other hand, prefer not to use RUMs to test for site reliability because:

- As the name implies, RUM requires user traffic. The site may be down, but since no user visited the monitored path, no alerts were triggered yet.
- Inconsistent Traffic and usage patterns make it hard to gauge for benchmarks.

Risks

Testing in production, in general, has a risk factor attached to it, which does not exist tests executed during CI/CD stages. Specifically, in synthetic monitoring tests, the following may affect the production environment:

- Corrupted or invalid data - Tests inject test data which may be in some ways corrupt. Consider using a testing schema.
- Protected data leakage - Tests run in a production environment and emit logs or trace that may contain protected data.
- Overloaded systems - Synthetic tests may cause errors or overload the system.
- Unintended side effects or impacts on other production systems.
- Skewed analytics (traffic funnels, A/B test results, etc.)
- Auth/AuthZ - Tests are required to run in production where access to tokens and secrets may be restricted or more challenging to retrieve.

Synthetic Monitoring Tests Frameworks and Tools

Most key monitoring/APM players have an enterprise product that supports synthetic monitoring built into their systems (see list below). Such offerings make some of the risks raised above

irrelevant as the integration and runtime aspects of the solution are OOTB. However, such solutions are typically pricey.

Some organizations prefer running probes on existing infrastructure using known tools such as Postman, Wrk, JMeter, Selenium or even custom code to generate the synthetic data. Such solutions must account for isolating and decoupling the probe's production environment from the core product's as well as provide monitoring, geo-distribution, and maintaining test health.

- Application Insights availability - Simple availability tests that allow some customization using Multi-step web test
- DataDog Synthetics
- Dynatrace Synthetic Monitoring
- New Relic Synthetics
- Checkly

Conclusion

The value of production tests, in general, and specifically Synthetic monitoring, is only there for particular engagement types, and there is associated risk and cost to them. However, when applicable, they provide continuous assurance that there are no system failures from a user's perspective. When developing a PaaS/SaaS solution, Synthetic monitoring is key to the success of service reliability teams, and they are becoming an integral part of the quality assurance stack of highly available products.

Resources

- Google SRE book - Testing Reliability
- Microsoft DevOps Architectures - Shift Right to Test in Production
- Martin Fowler - Synthetic Monitoring

Last update: August 22, 2024