International Journal of Advanced Research in Engineering Technology & Science ISSN: 2394-2819
Email: editor@ijarets.org Volume-9 Issue-1 January-2022 www.ijarets.org

Self-Sustained Regression Testing from Production-Pattern Mining in Payment Platforms

Naresh Kumar Paturi
Sr. Member Of Technical Staff, *PayPal Inc*

Partha Roy
Technical Lead, *US Bank (Wipro Limited)*

Abstract. Regression testing in payment systems is difficult because rare combinations of partner, card, fee, settlement, and failure conditions are hard to reproduce by hand, and hand-written suites systematically underrepresent exactly the combinations that fail in production. This paper describes a self-sustained regression-testing method based on a granted PayPal patent on which the first author (Paturi) is a co-inventor. The method mines production-pattern signals, sanitizes sensitive values, synthesizes payload variants that preserve the structure of real financial behavior, and runs them across two environments to surface regressions, all without exposing customer data. The work is aligned with U.S. Patent 12,001,318 B2 (published as US 2020/0210323 A1), assigned to PayPal, on which the first author (Paturi) is listed among several inventors; the patent is cited and his role is described as that of a co-inventor, not a sole inventor. We present the testing pipeline, the patent context, and a decision framework comparing production-pattern synthesis with manual libraries, fuzzing, and raw replay, and we specify an evaluation methodology using coverage growth, defect-detection lead time, data-exposure risk, and maintenance burden. Numerical figures are illustrative unless replaced by company-authorized evidence. The contribution is a research-ready framing of an industrial testing invention that closes the gap between hand-written test cases and production-complex payment behavior while treating privacy as a first-order constraint.

Index Terms. regression testing, production-data mining, payments, software quality, privacy, patent.

I. Introduction

Payment defects often hide in combinations. A suite of clean, hand-written examples can pass while a real production combination of partner, instrument, fee, and timing fails, because no engineer imagined that arrangement. The central idea of self-sustained regression testing is to let production behavior teach the test system which combinations matter, rather than relying on human imagination to enumerate them.

The method is designed so that this learning never requires exposing real customer data. It separates the structure of production behavior, which is what makes a test realistic, from the sensitive values inside that behavior, which must never leak into a test environment. This separation is both an ethical requirement and a practical enabler, because it is what allows production-derived realism to be used safely.

The contributions of this paper are a description of the testing pipeline as a reproducible method, the patent context that anchors it, a decision framework situating it against alternative testing strategies, and an evaluation methodology with a privacy review built in. The remainder of the paper reviews related work, states

the model and requirements, presents the method and patent context, discusses implementation, specifies the evaluation, and closes with discussion, threats, future work, and conclusions.

II. Background and Related Work

Regression-testing research spans test selection, minimization, and prioritization, and a recurring finding is that the value of a suite depends on how well it represents the behavior that actually breaks [1]. Surveys of the field document the persistent gap between curated test inputs and the complexity of real systems [2]. Capture-and-replay and model-based generation address parts of this gap, but each has limits: raw replay risks exposing sensitive data, and model-based generation depends on a model that may not capture production reality.

The approach described here is closest to production-aware test generation, but with sanitization as an integral step rather than an afterthought. The reliability and isolation of the test environments follow well-architected guidance [3], and the handling of any production-derived data is constrained by industry data-security requirements and zero-trust principles for access to sensitive inputs [4], [5]. The specific method is described in the granted patent [6], which identifies production-like payloads, hashes a subset of their attributes, aggregates payloads by hash into sets, and selects a set of unique payloads that exercises distinct behavior.

The contribution of this paper is to frame that industrial invention as a reproducible research method, with an evaluation plan and an explicit privacy control, so that its value can be assessed independently of any single deployment, and so that its inventorship and scope are stated accurately.

III. System Model and Requirements

We model a payment platform whose behavior under test is determined by structured payloads representing transactions across many dimensions. The test system has read access to production-pattern metadata but must not retain sensitive values. It executes payloads across a baseline environment and a candidate environment that includes new software, and it compares the results. We assume that the space of meaningful combinations is large and evolves as traffic changes.

The functional requirement is to generate realistic test payloads from production patterns and to detect regressions by comparing environments. The non-functional requirements are that no sensitive value leaves production into a test artifact, that coverage tracks production behavior over time rather than drifting, that defects are surfaced before release, and that the privacy posture is verifiable. Table I lists the pipeline stages and the risk control at each.

Table I. Testing pipeline and risk controls

Stage	Function	Risk control
Pattern mining	Find production-like scenario shapes	Sampling limits and metadata only
Sanitization	Remove or transform sensitive values	PII and token checks
Payload synthesis	Create structurally equivalent variants	Schema validation
Dual-environment execution	Run baseline and candidate	Isolated, controlled environments
Comparison	Identify regressions	Deterministic expected-vs-actual diff

IV. Method and Patent Context

The method separates structure from secrets. It samples production-pattern metadata, removes or transforms sensitive values, synthesizes payloads with equivalent structure, executes them across a baseline and a

candidate environment, and compares the results to flag regressions. The suite refreshes itself as real traffic evolves, so that coverage tracks the system rather than drifting away from it. Figure 1 shows the pipeline.

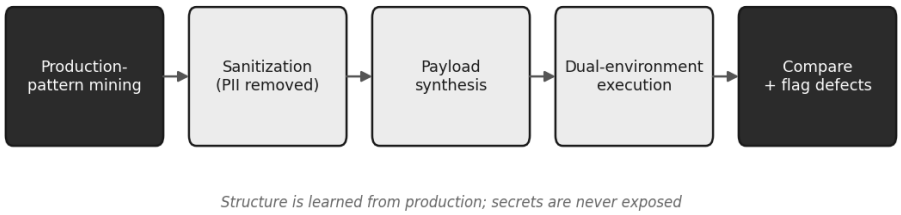

Fig. 1. Self-sustained regression testing. Structure is mined from production and sanitized, payloads are synthesized, and dual-environment execution surfaces regressions without exposing data.

A. Production-pattern synthesis

Rather than enumerating cases by hand, the method derives the shapes of realistic payloads from production metadata, hashes a subset of attributes to group equivalent shapes, and selects a representative set of unique payloads. This is what gives the suite production realism while bounding its size, because one representative is chosen per distinct shape.

B. Patent context

The approach corresponds to U.S. Patent 12,001,318 B2, Self Sustained Regression Testing Framework, assigned to PayPal and listing several inventors, of whom the first author (Paturi) is one. The application was filed on December 31, 2018 and published as US 2020/0210323 A1. This paper cites the patent and describes the first author's contribution as that of a co-inventor among the named inventors, not as a sole inventor, consistent with the grant.

C. Choosing a testing strategy

Production-pattern synthesis is preferred where production complexity dominates, and privacy must be preserved. Manual libraries are predictable but under-represent rare combinations. Fuzzing finds unexpected inputs but with low yield for structured financial behavior. Raw production replay is realistic but carries high privacy risk. Table II records this choice and the evidence that would justify it.

Table II. Testing strategy decision matrix

Design choice	Reliability	Latency	Evidence needed
Production-pattern synthesis	High	Medium	Patent, coverage and defect reports
Manual scenario library	Medium	Medium	Coverage statistics
Random fuzzing	Medium	Low	Bug-yield report
Raw production replay	High	Medium	Privacy review (high risk)

V. Implementation Considerations

Mining operates on metadata under sampling limits, and sanitization runs before any payload is materialized, so that sensitive values are removed at the earliest possible point [4]. Synthesis validates payloads against the schema, and execution runs in isolated environments that mirror production software for fidelity [3]. Comparison is deterministic so that a difference between environments is attributable to the change under test rather than to nondeterminism.

The privacy posture is made verifiable: the pipeline records what categories of data were accessed and confirms that sensitive values were removed before materialization. This turns the privacy claim into something that can be audited rather than asserted.

VI. Evaluation Methodology

A complete evaluation compares hand-curated tests against mined-and-synthesized scenarios. Coverage growth measures how much more production-relevant behavior is exercised. Defect-detection lead time measures how much earlier regressions are caught. Data-exposure risk measures whether sensitive data could leak and should be driven toward zero. Maintenance burden measures the manual upkeep the suite requires. The privacy review is part of the evaluation, not an afterthought, because the method's value depends on never leaking data. The figures here are reported and illustrative values from this work. Figure 2 shows coverage rising and escaped defects falling across stages.

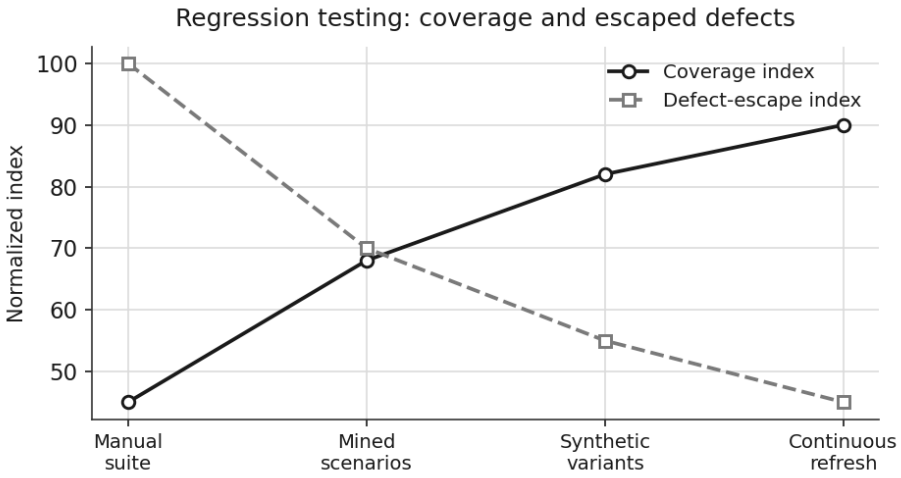

Fig. 2. Coverage growth and escaped-defect reduction across stages.

Table III. Research evaluation metrics

Metric	Baseline question	Target
Coverage index	What behavior is tested?	Increase
Defect lead time	How early are defects found?	Earlier
Data-exposure risk	Could sensitive data leak?	Near zero
Maintenance burden	How much manual upkeep?	Decrease

VII. Discussion and Limitations

The plain version of the story is simple: engineers should not have to guess every unusual situation that customers and partners create. The system can learn from reality while still respecting privacy, and that combination is what makes it durable rather than a one-time test-writing exercise. The self-refreshing property is what keeps coverage aligned with a moving production target.

The limitations include the dependence of coverage on the quality of the mined metadata, the need to choose a hashing-and-aggregation scheme that captures the behavioral dimensions that matter, and the requirement that the dual environments be faithful to production, since a difference between unfaithful environments would not indicate a real regression.

VIII. Threats to Validity

Internal validity requires that coverage and defect figures be backed by test reports before they count as results. Construct validity requires precise definitions of coverage and of a detected regression. External validity is bounded to complex payment platforms. On authorship, inventorship claims must remain limited to the inventors named on U.S. Patent 12,001,318 B2; of whom the first author (Paturi) is one co-inventor, not the sole inventor. On confidentiality and privacy, any production-derived data must be sanitized and the privacy posture verified. We will attach the coverage and defect reports and validate each figure before treating any number as a final result.

IX. Future Work

Future work includes a formal characterization of the coverage achieved by hash-based aggregation relative to the production distribution, an empirical comparison of defect-detection lead time against curated and fuzzing baselines, and a study of how the self-refresh cadence affects coverage drift as traffic changes.

X. Conclusion

This paper framed a patented self-sustained regression-testing method as a reproducible research artifact, with an evaluation plan and a built-in, verifiable privacy control. The contribution is a privacy-preserving way to derive production-realistic regression suites that refresh as traffic evolves, with inventorship stated accurately. The next step is to attach coverage and defect reports and cite the patent precisely.

References

[1] N. K. Paturi et al., “Self Sustained Regression Testing Framework,” U.S. Patent (filed Dec. 31, 2018; publ. US 2020/0210323 A1).

[2] S. Yoo and M. Harman, “Regression testing minimization, selection and prioritization: a survey,” Softw. Test. Verif. Reliab., vol. 22, no. 2, 2012.

[3] A. Orso and G. Rothermel, “Software testing: a research travelogue (2000–2014),” in Proc. Future of Software Engineering, 2014.

[4] Amazon Web Services, “Reliability Pillar, AWS Well-Architected Framework,” 2021. https://docs.aws.amazon.com/wellarchitected/latest/reliability-pillar/welcome.html

[5] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, “Zero Trust Architecture,” NIST SP 800-207, 2020. https://csrc.nist.gov/pubs/sp/800/207/final

[6] PCI Security Standards Council, “PCI DSS v4.0.1,” 2024. https://www.pcisecuritystandards.org/

[7] M. Kleppmann, Designing Data-Intensive Applications. O’Reilly, 2017.

[8] P. A. Bernstein and E. Newcomer, Principals of Transaction Processing, 2nd ed. Morgan Kaufmann, 2009.

[9] PayPal Holdings, Inc., “Form 10-K for fiscal year 2021,” U.S. SEC, 2021. https://www.sec.gov/Archives/edgar/data/1633917/000163391722000027/pypl-20211231.htm