Penetration test vs bug bounty.

A penetration test is a contracted, time-boxed engagement against a defined scope, delivered by a known team, with a written report. A bug bounty programme pays outside researchers per valid finding under a public or invite-only ruleset, with the work happening continuously and the coverage depending entirely on who shows up. They cover different ground. Most mature programmes run both: pentests for periodic depth and assurance, bug bounty for ongoing breadth.

HOW IT WORKS

01 How do they actually differ?

Key differences in practice:

- Coverage. A pentest commits to testing the scope thoroughly. A bug bounty has no coverage guarantee. You get findings from whoever decides to look at your programme.
- Cost model. A pentest is project-priced. A bug bounty pays per finding, plus platform fees. A quiet bounty costs little; a busy bounty can cost much more than a pentest.
- Timing. A pentest happens on a schedule. A bug bounty runs continuously.
- Researcher quality. A pentest team is known and contracted. Bug bounty researchers self-select. Most reports are low-quality; the high-quality ones make the programme worth it.
- Triage burden. A pentest report is curated. A bug bounty produces a steady stream of duplicate, out-of-scope, and low-severity submissions that someone has to triage.
- Compliance evidence. Auditors usually accept pentest reports. Most do not yet accept bug bounty output as equivalent.
- Liability and rules of engagement. A pentest contract specifies what the tester can and cannot do, with indemnity. Bug bounty terms place the legal burden on the researcher accepting the programme rules.

02 When does each one fit?

A pentest fits when you need:

- Assurance before a launch, audit, board review, or procurement gate.

SOURCES

[1] NIST SP 800-115

[2] ISO/IEC 29147 Vulnerability Disclosure

[3] CISA Coordinated Vulnerability Disclosure Process

- Coverage commitments against a defined scope.
- A contractually agreed deliverable.
- A report auditors will accept.
- A predictable cost.

A bug bounty fits when you have:

- A mature security programme that can triage incoming reports without slowing engineering.
- A scope that benefits from ongoing attention from many different researchers.
- The budget to absorb variable payouts (and the cap mechanisms in place).
- A response process that meets the researcher community's expectations on time-to-acknowledge and time-to-fix.

Bug bounty programmes that launch before the basics are in place do more harm than good: researchers find low-hanging fruit faster than the team can fix, the report queue grows, and the relationship sours.

03 Should we run both?

Most mature programmes do, in this order.

1. Internal hygiene first: continuous vulnerability scanning, secure-SDLC controls, automated code review. 2. Periodic external penetration testing for depth and compliance evidence. 3. Bug bounty programme once the easy issues are out of the way and triage capacity exists.

Launching bug bounty before steps 1 and 2 is the most common avoidable mistake. The programme produces reports faster than the team can respond, the researchers get frustrated, and the brand suffers.

04 Should we use a bug bounty platform or run it ourselves?

Platforms (the major bug bounty marketplaces) provide triage, payment processing, researcher community, and reputation systems. They take a fee. For most organisations the fee is worth it. Self-managed programmes work when the security team is large enough to handle every part of the process, including legal review, payment infrastructure, and researcher communications.

Choose pentest, bug bounty, or both with confidence.

securelayer7.net/learn/pentest/pentest-vs-bug-bounty

Open online