

Forum:[Hack Forums](#)

Thread: [BUG HUNTING MUST HAVE TOOL](#)

coocochi - September 12, 2026, 8:12 am

I came across an interesting open-source security tool called BugScanner

What is BugScanner?

BugScanner is a Python-based reconnaissance and web vulnerability assessment framework designed for authorized penetration testing, bug bounty research and defensive security assessments.

Rather than focusing on just one vulnerability class, it attempts to combine reconnaissance and vulnerability assessment into a single workflow.

Reconnaissance

The project can perform several types of discovery, including:

- Subdomain enumeration
- DNS discovery
- TCP port scanning
- Service/banner identification
- Technology fingerprinting
- Web endpoint discovery
- Discovery of common API, authentication, administration and backup paths

Vulnerability Assessment

BugScanner includes modules aimed at identifying or verifying issues such as:

- Reflected XSS
- SQL injection
- CORS misconfigurations
- SSRF
- Open redirects
- JWT security weaknesses
- IDOR/path tampering
- Sensitive information exposure

It can also integrate with Nuclei when Nuclei is installed, allowing its results to be incorporated into the overall assessment.

Interesting feature: adaptive scanning

One of the more interesting aspects of the project is its adaptive request handling.

There is a token-bucket rate limiter that can react to responses such as `429 Too Many Requests` and `503 Service Unavailable`, together with WAF detection and jitter controls.

The goal is to make automated assessment less noisy and reduce unnecessary request pressure.

Reporting

The project can generate structured JSON output as well as HTML reports containing vulnerability findings and CVSS v3.1 severity information.

This makes it useful not only for discovering potential issues but also for organizing findings for later manual verification.

How it is structured

The project uses an asynchronous Python architecture built around `asyncio` and `httpx`.

The repository separates functionality into areas such as:

Recon

- +-- Subdomain discovery
- +-- Port scanning
- +-- Technology fingerprinting
- +-- Endpoint discovery

Vulnerability assessment

- +-- XSS
- +-- SQL injection
- +-- CORS
- +-- SSRF
- +-- Open redirect
- +-- JWT
- +-- IDOR
- +-- Information disclosure

Core

- +-- HTTP client
- +-- Rate limiting
- +-- Data models / scoring
- +-- Report generation

Getting started

The project currently targets Python 3.11+

A basic setup looks like:

On powershell

```
git clone https://github.com/eldarshiraliyev/BugScanner.git  
cd BugScanner
```

```
python3 -m venv .venv  
source .venv/bin/activate
```

```
pip install -r requirements.txt  
python cli.py --help
```

For a first test, I'd recommend using a deliberately vulnerable lab or another target where you have explicit authorization.

Start with reconnaissance:

```
python cli.py scan https://target-domain --mode recon
```

The project also documents full assessment, vulnerability-only scanning, custom port ranges, request rate configuration and JSON reporting.

Important

This is an active security-testing tool.

Do not point it at random websites or infrastructure that you do not own or have permission to test.

Only use it against:

- Your own applications
- Authorized penetration-testing targets
- Bug-bounty assets explicitly listed as in scope
- * Deliberately vulnerable training environments

Automated scanning can generate significant traffic and may trigger security controls or disrupt poorly configured systems.

My take

What I find interesting about BugScanner is the attempt to combine reconnaissance, fingerprinting, vulnerability verification, rate control and reporting into one Python framework rather than requiring a completely separate tool for every stage.

It's also a good project to study if you're interested in understanding how automated web-security tooling is assembled: asynchronous HTTP requests, DNS enumeration, service discovery, technology detection, vulnerability checks and report generation are all separated into different components.

Full credit to the developer: "eldarshiraliyev" for creating and maintaining the project.

<https://github.com/eldarshiraliyev/BugScanner>

If you're experimenting with it, I'd recommend starting inside a legal lab environment and manually validating anything the scanner reports rather than treating automated findings as confirmed vulnerabilities.