

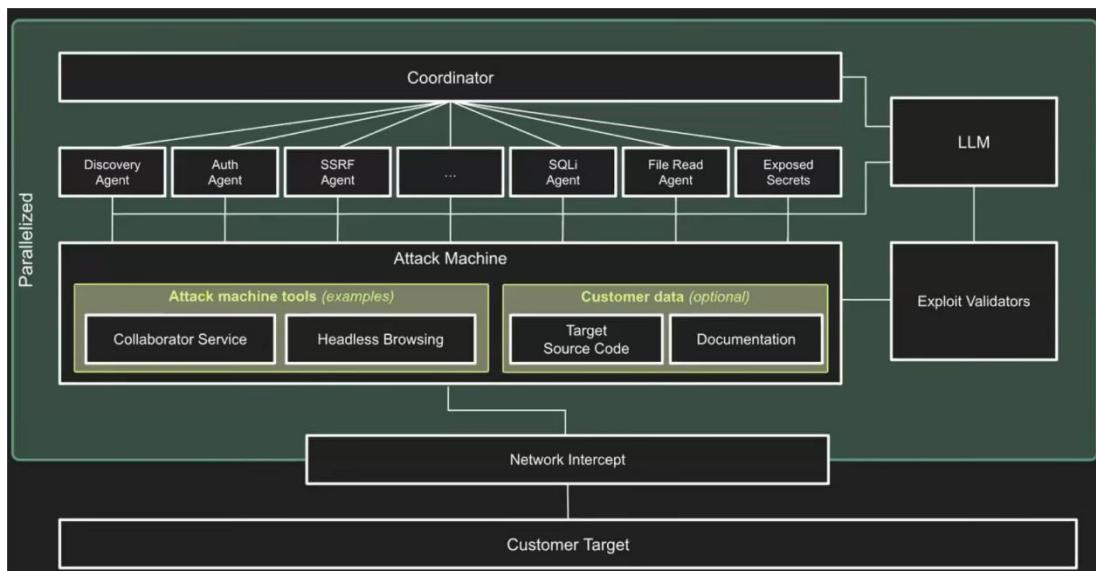
1. Recent Summary

Over the past week, I have looked into Xbow's official website and watched 10 YouTube videos. Most of these videos have content from Xbow's team members, and a small number are third-party reviews. Through these materials, I got a clear idea of how Xbow works. I also did a short study on three AI-driven penetration testing projects: PентestGPT, HexStrike and Burp AI. During my research on Xbow, I found many interesting things.

Xbow's official blog has a lot of technical details about different vulnerabilities. But I was especially interested in the product's demo and overall structure. Instead of getting stuck on small technical details, I found a live demo shared by Xbow's Security Lead on August 26, 2025. This demo helped me get a basic understanding of the product's overall structure and key functions.

2. Xbow Architecture and Demonstration

As the diagram shows, Xbow works because the following parts work together:



Coordinator: It manages the penetration testing process, like a manager in a penetration testing team. The Coordinator sets the testing methods and what's more important, and gives tasks to different Agents.

Agents: These are different kinds of AI-based tools that do specific detection and attack tasks. They are made based on “vulnerability primitives” (a technical term for basic vulnerability units).

Example 1: Discovery Agent – Usually, the first step of testing is to map the “attack surface” (all parts of a system that could be attacked). The Coordinator tells the Discovery Agent to list domains, endpoints, parameters and other related things.

Example 2: Authentication Agent – After the discovery step, it tries to log in (if possible). Once

enough information about the attack surface is collected, the Coordinator gives tasks to multiple Agents so they can check for vulnerabilities at the same time.

Exploit Validators: When an Agent finds a possible vulnerability, it “raises a flag” (reports it) and sends the result. Then a Validator checks the result again. There are different types of Validators: some use AI, while others use simpler methods. For example, an XSS Validator opens the URL in a “headless browser” (a browser without a visual window) and checks if JavaScript runs. This helps confirm if the vulnerability is real.

All tasks done by each Agent run in an environment called an “attack machine”. This machine has many tools, including ones made by Xbow and common penetration testing tools. Xbow also added a “collaboration service” and a headless browser to handle complex JavaScript apps. The browser analyzes and interacts with very dynamic frontends (the part of an app users see), which makes attack surface mapping and testing more complete.

The last key part is the Network Intercept module, which has two main jobs:

- 1) It records all requests and responses that Xbow sends. After the penetration test finishes, users get a full record of the network interaction.
- 2) It works as a protection tool (like a firewall). Users can set rules here to block access to certain domains. This stops Xbow from targeting assets that are not in the test scope (such as production domains). It is one of the many security protections for clients.

In the demo that followed, three types of assessments were shown:

- A full evaluation of the application
- Focused testing on specific threat paths
- Testing again for vulnerabilities found before

The screenshot shows the Xbow Demo interface. At the top, there's a navigation bar with 'XBOW Demo' and 'Akamai CloudTest'. Below that is a progress bar with five steps: 'Assessment type' (highlighted in grey), 'Target configuration', 'Configuration check', 'Run assessment', and 'Review results'. The main area is titled 'Choose an assessment type'. It contains three options:

- Comprehensive application assessment**
Full security assessment across the entire application
Specific feature / release details (Optional)
[Text input field]
- Test for specific threat vectors**
Focus on particular vulnerability classes
- Retest previous vulnerabilities**
Verify fixes and remediation efforts

On the right side, there's an 'Assessment summary' section with the following details:

Application	Akamai CloudTest
Available credits	11377
Assessment type	Comprehensive
Target URL	https://clouddtestmanager.soasta.com
Rate and timing	60 reqs/sec limit, off-business hours
Attackable domains	clouddtestmanager.soasta.com, *.clouddtestmanager.soasta.com

At the bottom right are 'Cancel' and 'Continue' buttons.

Next, users need to set the required target URLs and some optional settings. Optional settings include login details, custom headers, rate limits and so on.

<p>Target configuration</p> <p>How XBOW accesses and assesses your application</p> <p>Target URL https://cloudtestmanager.soasta.com</p> <p>+ Add credential set</p> <p>> Specific authentication instructions (optional)</p> <p>Source code and documentation</p> <p>Upload source code, documentation, or other relevant files</p> <p>Select files</p> <p>> Custom headers (optional)</p> <p>X-Bug-Bounty <input type="text"/> HackerOne-xbow X</p> <p>+ Add header</p> <p>> Rate limit and allowed window (optional)</p> <p>> Safety rules (optional)</p> <p>> Canaries (optional)</p>	<p>Application Akamai CloudTest</p> <p>Available credits 11377</p> <p>Assessment type Comprehensive</p> <p>Target URL https://cloudtestmanager.soasta.com</p> <p>Rate and timing 60 reqs/sec limit, off-business hours</p> <p>Attackable domains cloudtestmanager.soasta.com, .cloudtestmanager.soasta.com</p>
---	--

After setting these up, clicking “Start / Run Assessment” starts the test. First, the system does “pre-test checks”: it verifies if it can connect to the target, makes sure login details work, and does a simple crawl to roughly set the app’s test scope. This step takes about 10 to 15 minutes.

<p>Assessment type</p> <p>Target configuration</p> <p>Configuration check</p> <p>Run assessment</p> <p>Review results</p>	<p>Assessment summary</p> <p>Application Akamai CloudTest</p> <p>Available credits 11377</p> <p>Assessment type Comprehensive</p> <p>Target URL https://cloudtestmanager.soasta.com</p> <p>Rate and timing 60 reqs/sec limit, off-business hours</p> <p>Attackable domains cloudtestmanager.soasta.com, .cloudtestmanager.soasta.com</p>
---	---

Configuration check

Confirm the parameters for how XBOW will run the assessment

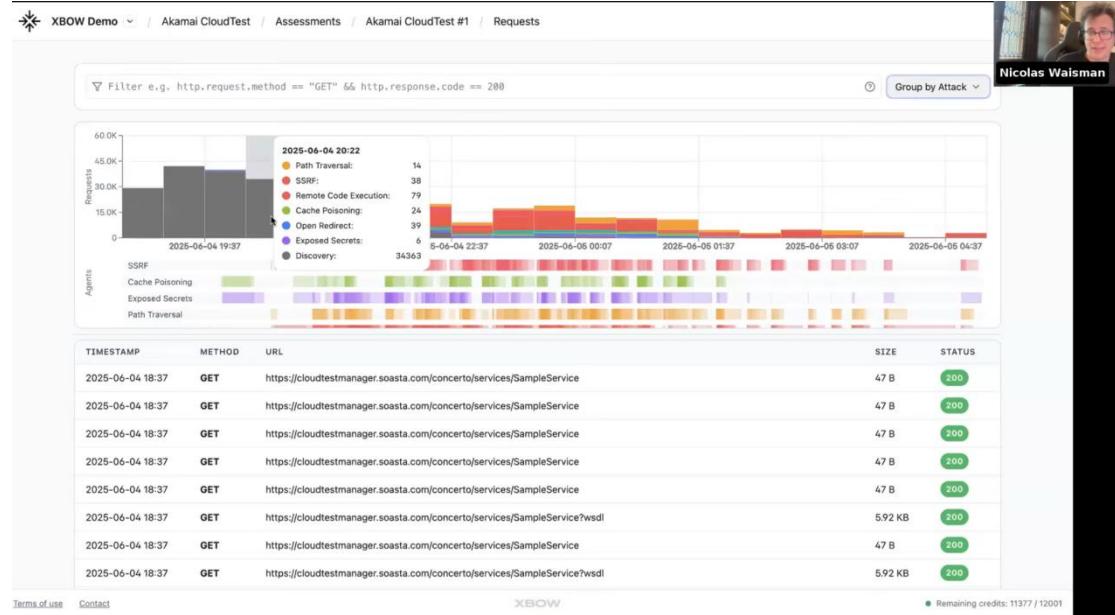
Status

- ✓ Connected target is reachable at <https://cloudtestmanager.soasta.com>
- ✗ Skipped no credential sets
- ⠋ Crawling... discovering scope...

[Back](#) [Schedule later](#) [Run assessment](#)

The assessment screen shows real-time progress and a chart. By default, the chart is grouped by status (but it can also be grouped by attack type). At first, the process focuses on discovery. As

testing goes on, many attack attempts and exploitation actions start, and discovery work slowly decreases. Importantly, even during the attack step, the system may sometimes find new endpoints and add them to the testing list. So discovery items may still show up from time to time in the progress updates.



When “Findings” (vulnerability results) are generated, Xbow lists each finding in a table. Clicking any entry shows the following details:

- A summary of the vulnerability and how serious it is
- Full step-by-step guides to exploit or reproduce the vulnerability (how to trigger it)
- An explanation of the vulnerability’s impact and suggestions to fix it
- “Proof of concept” (real evidence that Xbow can trigger the vulnerability, such as packet records or output logs)

This screenshot shows the Xbow Demo interface with the 'Assessment' and 'Findings' sections visible. The 'Assessment' section on the left shows 'Akamai CloudTest #1' status as 'Test successful' with a green checkmark. It also includes a link to 'Explore findings below'. The 'Control' section on the right shows the test was 'Started by Andy' on '2025-06-04 18:30'. The 'Findings' section below lists seven vulnerabilities with details like ID, Name, Path, Severity, Status, and Last Seen. The 'Overview' section at the bottom provides a summary with metrics: 22.2K Commands, 342.1K Requests, and 249 Endpoints.

3. Understanding Xbow's Team Structure

Using the architecture diagram and a past interview with Xbow's CEO, I learned that the company's tech department is split into three teams:

- Security Team: In charge of security research
- AI Team: Works on developing algorithms
- Engineering Team: Takes care of coding and putting the product into use

This team structure clearly divides the responsibilities of each team.

Founders of Copilot and GitHub Advanced Security

Nicolas Waisman

Oge de Moor Founder & CEO	Nico Waisman Head of Security	Albert Ziegler Head of AI	Andy Rice Head of Engineering	Aqeel Siddiqui Head of Product and Customer Success
Prior: <ul style="list-style-type: none">• Founder of Copilot• Founder of GitHub Next• Founder & CEO of Semmle• Professor at Oxford	Prior: <ul style="list-style-type: none">• CISO of Lyft• Blackhat Review Board• R&D at Semmle• VP at Immunity• Legendary hacker	Prior: <ul style="list-style-type: none">• Copilot co-founder• AI researcher at GitHub Next• R&D at Semmle• Author of acclaimed textbook on prompting and designing AI applications	Prior: <ul style="list-style-type: none">• Copilot team• Java Compiler team, Google• Professor at Cambridge	Prior: <ul style="list-style-type: none">• Copilot team• Technical program manager, GitHub/Microsoft• Director, Services Operations, Veracode

4. Xbow's Methodology

Later, I looked at Xbow's presentation at DEF CON on October 17, 2024. The presentation included a demo and a general introduction to their working method.

Our Approach

- Thousands of Benchmark CTF challenges
- Fully-autonomous agent
- Great focus in web application challenges
- Different levels of difficulty: easy, medium and hard
- Attack machine with pentesting tools
- Exclusive set of 104 Novel Benchmarks for validation

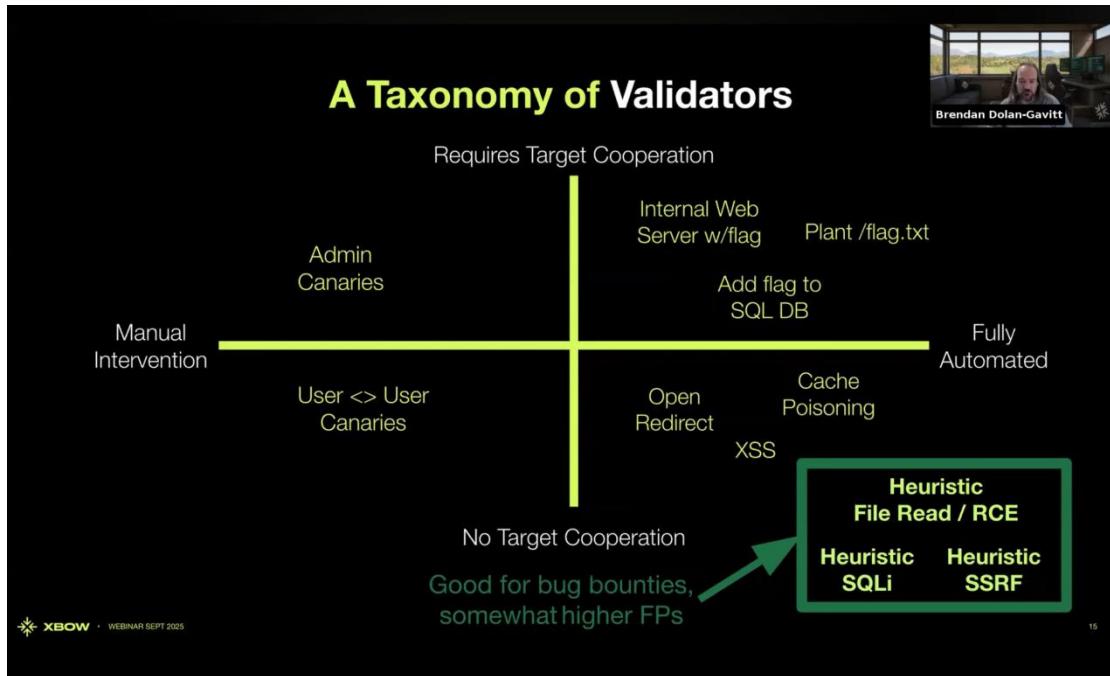
DEFCON ➤➤➤

During their research, Xbow used the following five papers and made improvements based on what the papers found:

- An Empirical Evaluation of LLMs for Solving Offensive Security Challenges
- LLM Agents can Autonomous Hack Websites
- NYU CTF Bench: A Scalable Open-Source Benchmark Dataset for Evaluating LLMs in Offensive Security
- Teams of LLM Agents can Exploit Zero-Day Vulnerabilities
- LLM Agents can Autonomous Exploit One-day Vulnerabilities

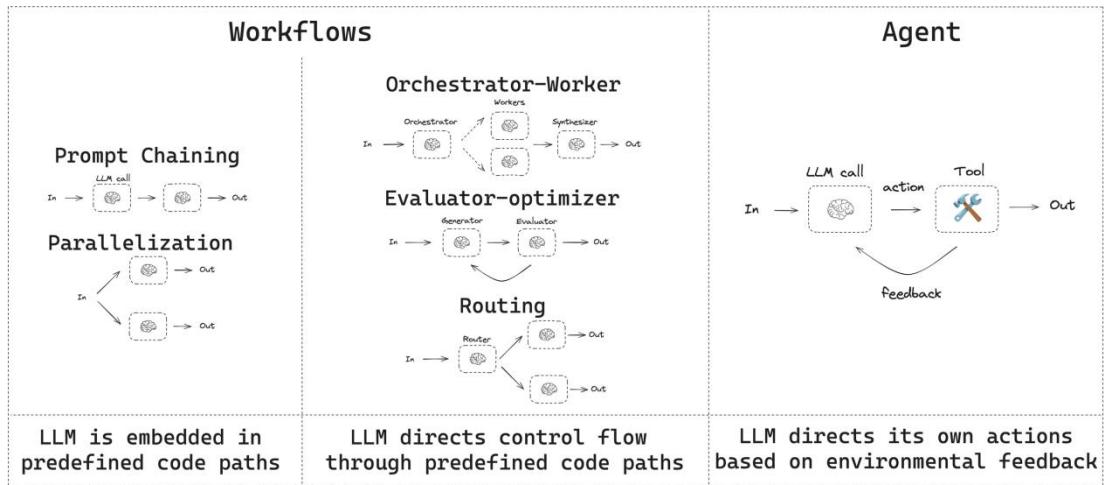
5. Zero False Positives

On October 1, 2025, Brendan Dolan-Gavitt gave a presentation called 200 Zero Days 0 False Positives. In this talk, he explained Xbow's more complex system structure and how the Validator part is used to lower the number of "false positives" (results that wrongly say a vulnerability exists).

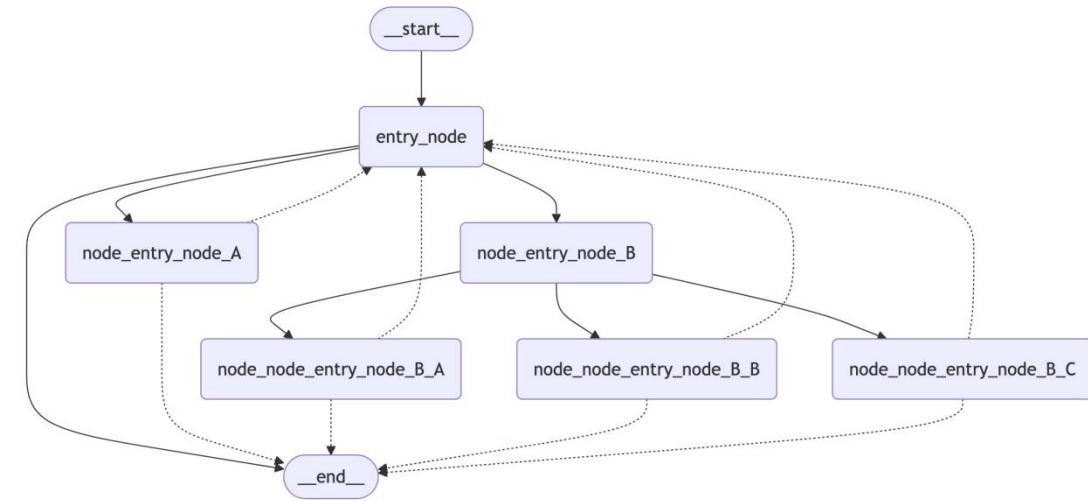


6. LangGraph

An analysis of Xbow's system architecture and Validators reveals that Xbow is built using LangGraph.

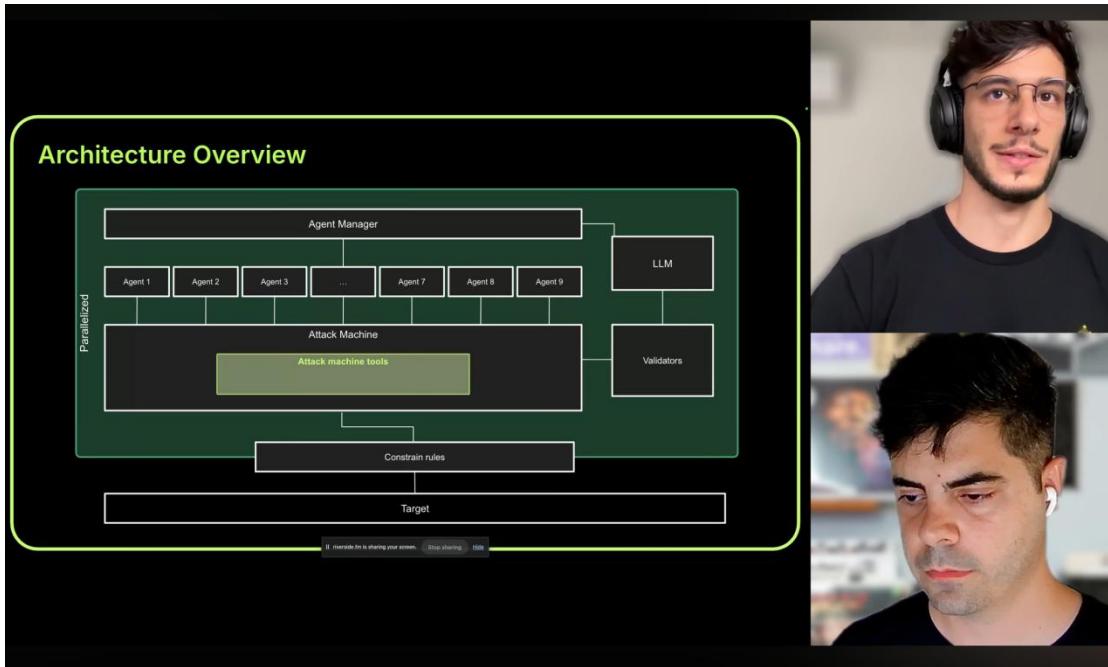


The following outlines LangGraph's workflow and Agents: The different Agents in Xbow are essentially distinct nodes within LangGraph. LangGraph orchestrates the collaborative logic between these Agents.



7. Building a Minimal Viable AI Penetration Testing System

Xbow's lecture titled [Prompt. Scan. Exploit - AI's Journey Through Zero-Days And A Thousand Bugs](#), analyzed on September 4, 2025, provides more specific guidance for building a minimal viable AI penetration testing system.



Xbow's development process has gone through four phases. They encountered challenges during the "Zero-Day Hunting" phase (Phase 3) and are currently in Phase 4: "Vulnerability Bounties."

Step 3: Zero-Day Hunting

Focused on OSP before targeting Real Bounty Programs

- More realistic than CTFs: realistic apps, complexity, real-world bugs *
- Tech diversity: broad exposure to frameworks and stacks
- Docker! "Easier" to deploy
- Access to source code (Blackbox vs Whitebox)
- LLM-powered automation: generation of compose files

zero-days	Published CVEs	Pending CVEs
174	22	152

* *Bounty tip:*
Bugs identified here can have a direct impact on Bug Bounties (specific targets)

If riverdale is sharing your screen. Stop sharing. Hide



Evolution

From CTF to Real-World Hunting

```

graph LR
    1[Step 1: CTF Challenges  
Portswigger and Pentesterlab environments] --> 2[Step 2: Custom Benchmarks  
Real-world scenarios, never seen by LLM]
    2 --> 3[Step 3: Zero-Day Hunting  
White-box testing of open source projects]
    3 --> 4[Step 4: Bug Bounty Targets  
Black-box testing in production environments]
    style 4 fill:#00ff00,color:#fff
    
```

2024
2025

Vulnerability Classes

Our approach vs Bug Bounty Approach

> Remote Code Execution Critical system compromise vulnerabilities	= SQL Injection Database access and manipulation	= External XML Entities (XXE) XML processing exploitation vectors
</> Cross-Site Scripting Client-side code execution	= Server-Side Request Forgery Internal network access	= Path Traversal Unauthorized file system access
= Server-Side Template Injections Template engine exploitation	= Information Disclosure Sensitive data exposure	= Cache Poisoning Cache manipulation attacks
= Open Redirects URL redirection abuse	= IDORs and Business Logic Authorization/Authentication and business logic attacks	

The lecture outlines the architecture of a minimal viable system and identifies specific challenges, along with corresponding mitigation strategies. These challenges include:

- 1) High AI scanning costs and insufficient profitability
- 2) A massive number of URLs requiring efficient filtering and deduplication
- 3) Difficulty in conducting in-depth testing, leading to "blind scanning"
- 4) Loss of testing scope control (e.g., out-of-scope targeting, excessive post-exploitation)
- 5) High false positive rates and "cheating" behaviors in LLMs
- 6) Performance limitations of single models, failing to cover all vulnerability scenarios
- 7) Unreasonable scan configurations resulting in resource waste and low efficiency

References

Finding Web App Vulnerabilities with AI

<https://youtu.be/v-McepNOrTQ?si=UC7kx0lTypyLYjoV>

April 16th, 2025

[SSTF2024] Invited Talk 1 : AI Agents for Offensive Security

<https://youtu.be/ncwjfg19EGw?si=BtruwIYRSAZ1m6hu>

November 7th, 2024

Prompt. Scan. Exploit - Ai's Journey Through Zero-Days And A Thousand Bugs

https://youtu.be/y_aQQmDMAY4?si=gi64usPtVA4fiK5S

September 4th, 2025

DEF CON 32 - Leveraging AI for Smarter Bug Bounties - Diego Jurado & Joel Niemand Sec Noguera

<https://youtu.be/8IzKwczq0J?si=kbe-igaYntT3tR9D>

October 17th, 2024

Autonomous Hacking? This Startup May Have Just Changed Penetration Testing Forever

<https://youtu.be/VeOMYBSk3Dk?si=aKKOYQxh4va6f6li>

July 7th, 2025

XBOW - AI Hacking Agent and Human in the Loop with Diego Jurado (Ep. 134)

<https://youtu.be/rvA8IbyogJ0?si=CRy5eHnMLflq2bEO>

August 4th, 2025

Cracking the Code on Offensive Security With AI ft XBOW CEO and GitHub Copilot Creator Oge de Moor

https://youtu.be/9mlphDV9m9c?si=XtB_Oka9H-T1AGJ3

December 10th, 2024

200 Zero Days 0 False Positives

<https://www.youtube.com/watch?v=WduURDgUg4E>

October 1st, 2025

Live Demo: XBOW in Action

<https://youtu.be/CyJi9LEIG4E?si=cmCmgvPZxO2NQn7C>

August 26, 2025

Leveraging AI for Smarter Bug Bounties | Bug Bounty Village, DEF CON 32

<https://www.youtube.com/watch?v=YDsHI2acEVA>

April 22, 2025