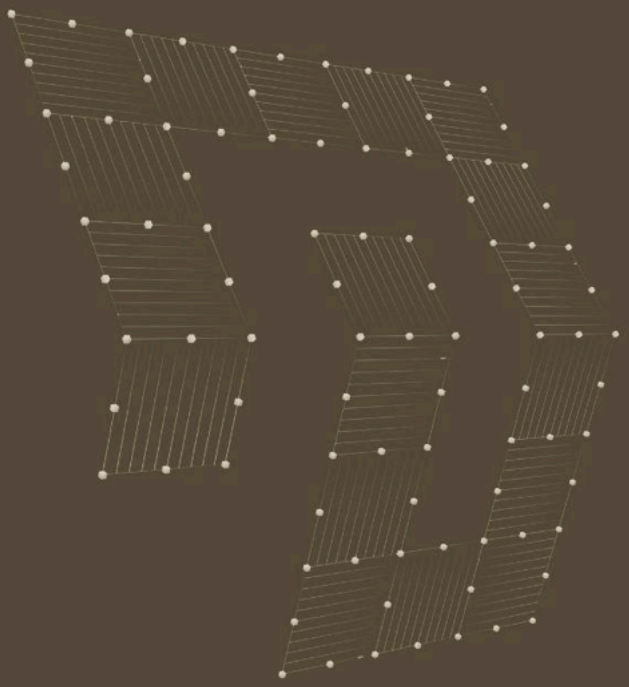


Comparing AI Application Security Testing Platforms

Aikido vs. XBOW

Luca Carettoni & Anthony Trummer

DOYENSEC

CONTENT

INTRODUCTION

Doyensec performed a side-by-side comparison of two AI-powered penetration testing platforms - Aikido's Attack AI Pentest and XBOW's Lightspeed in order to evaluate their abilities to properly identify vulnerabilities in modern web applications. This included manually validating all findings and classifying them as either true positives or false positives. Additionally, we looked at their overall testing process, including the configuration, impact on tested applications, quality and content of the reports, cost, and speed.

Keywords

artificial intelligence (AI), penetration testing, application security, code review, Aikido, XBOW

Context

Application security auditing has historically been performed through a combination of automated static and dynamic analysis (i.e., SAST and DAST) along with manual human code review and dynamic testing.

With ominous headlines such as the supposed power of Anthropic's Mythos constantly in the news, Artificial Intelligence (AI) is widely thought of as only recently coming into its own. However, those in the cybersecurity industry know it's been present and evolving for years. After all, it's been nearly 10 years since the first *Cyber Grand Challenge* (CGC). In that contest, sponsored by the United States' Defense Advanced Research Projects Agency (DARPA), competitors were able to accomplish the mission to *"...create automatic defensive systems capable of reasoning about flaws, formulating patches… in real time*". AI's appeal for defending software is clearly not a new concept and this type of technology would be invaluable to organizations that create and maintain applications. Even with automation, cybersecurity teams within organizations struggle to keep up with the increased speed of today's development processes, code complexity, and constant cybersecurity attacks - some of which are using AI themselves.

In today's environment of rapidly maturing AI technology, several platforms have emerged to address this need. These platforms broadly claim to be able to compete favorably with the traditional comprehensive style of application security assessments. Some go as far as to say their AI testing will be simultaneously better, cheaper, and faster.

Regardless of our opinions on that topic, in this paper, we don't focus on the human vs. AI comparison. Rather, **we take a look at the relative performance of two widely available commercial platforms (namely Aikido and XBOW) to examine the state of the art and potentially inform organizations that might be considering AI-based solutions for web security testing.** While the results are platform-specific, several considerations around the use of LLM-based solutions for vulnerability discovery can be applied to other vendors as well.

Motivation

Why did Doyensec decide to perform this analysis? We received a great deal of positive feedback on our post comparing SAST tools (Semgrep vs. CodeQL). The majority of which, not surprisingly, was from teams that were at a crossroads themselves in determining which

technology to move forward with. So, we assumed many organizations were similarly at this point in evaluating the potential use of AI tools and needed an unbiased opinion.

Further, as a boutique application security consultancy, we were also curious about how the adoption of AI will impact the future of testing. To understand the current maturity levels of these AI platforms, it was necessary for us to put some vendors' claims to the test.

Disclaimer: *In the spirit of full transparency, our evaluation was sponsored by Aikido (Aikido Security BV). That said, they were in no way involved in the collection or presentation of these results, but did have input into the constraints mentioned later in this paper. We purposefully separated the collection of data from its analysis (i.e., different people did each) and are confident that neither was biased towards either solution in any way. We will publish as much data as we can about the testing, so anyone inclined to do so can validate our methodology and results themselves.*

Scope

The scope of this comparison was limited to the head-to-head comparison of the aikido/attack AI pentest platform and the XBOW platform. This comparison was specifically not intended to evaluate performance against human testers, nor against current traditional SAST applications. XBOW was chosen as a competitor, partly due to its receiving a good amount of publicity and piquing our curiosity into wanting to understand how well the AI itself functioned versus how much manual testing/validation might be going on behind the scenes, to the extent possible. From both a capabilities and pricing perspective, the two platforms appeared broadly comparable, making a direct head-to-head comparison reasonable.

Goals

In our examination, we primarily wanted to compare the number of true and false positives provided by each platform. We purposely omitted examining true and false negatives from the goals. Since this isn't a contrived benchmarking code, there would be no guarantee we had found all the vulnerabilities, even if we performed an exhaustive and very time-consuming code review. Further, this allowed us to avoid an AI vs. human comparison, which wasn't the objective. Additionally, our supplementary goals included attempting to ascertain the platforms' capabilities, thoroughness, workflows, and the costs involved.

Verification

Each reported finding from the platforms was manually validated by a qualified member of the Doyensec team. This was done not just to validate that the findings were legitimate, but also to confirm their severities, based on our experience and expertise. A peer review was also conducted to minimize the risk of human error.

BACKGROUND

Constraints

Based on our experience with previous research, we deliberately attempted to make the process as straightforward as possible. We also needed to take the financial impacts of paying for the testing into account, where applicable.

To that end, we limited the analysis to only **two applications**. Due to the limited sample size, we cannot say that the results are generalizable, but we hope they are sufficiently informative nonetheless.

Lastly, it should be understood that any configurations/settings not explicitly mentioned should be assumed to be at either their defaults or in an obvious state, based on the context.

TARGET APPLICATIONS

Criteria

Wanting to emulate a real-world scenario as much as possible during testing, we decided to limit the candidates to OSS self-hostable applications. This way, we would have complete and fully functional applications, together with real-life source code. Further, we wanted to ensure they were of sufficient complexity to be interesting from a security perspective and with realistic attack surfaces and functionalities (e.g., public and private components, cross-user interactions, multiple authentication mechanisms).

To ensure the possibility of authorization vulnerabilities, the targets needed to be multi-user web applications.

Lastly, to guarantee support by the platforms and to reflect commonly used modern languages, we only selected applications written in one of these programming languages: Python, PHP, Ruby, Go, JavaScript, or TypeScript.

Curated Candidate Pool

Our first step in narrowing down potential candidates was to use the awesome-selfhosted list of approximately 1092 repositories of self-hosted applications from GitHub. We felt this would provide a better signal/noise ratio than a random GitHub search.

In the spirit of the exercise, we vibe-coded a ready-to-use Python script that would:

- Use the *awesome-selfhosted* list (linked above) as the source
- Extract the GitHub repositories
- Filter for Python, PHP, Ruby, Go, JavaScript, TypeScript
- Detect deployability (Dockerfile, docker-compose, helm, k8s)
- Detect the frameworks
- Save everything to a CSV dataset
- Use a GitHub token
- Include throttling and retries
- Run in ~2–4 minutes, depending on the API speed

This resulted in a CSV containing a list of 442 web applications.

Randomized Selection

To avoid bias, Doyensec chose two applications at random from the curated list of 442. This was done without any input from any third parties (i.e., the vendors whose platforms were being tested).

To select the two applications, we executed `` `echo $(($RANDOM%442))` `` twice as shown below. As a result, we used rows 203 and 365 from the CSV file.

```
ikki@ip-192-168-68-53 ~ % date
Mon Mar 16 16:25:54 CET 2026
ikki@ip-192-168-68-53 ~ % echo $(($RANDOM%442))
203
ikki@ip-192-168-68-53 ~ % date
Mon Mar 16 16:25:57 CET 2026
ikki@ip-192-168-68-53 ~ % echo $(($RANDOM%442))
365
ikki@ip-192-168-68-53 ~ % date
Mon Mar 16 16:26:00 CET 2026
```

Selected Applications

The applications selected for this comparison are the following:

Fider (https://github.com/getfider/fider) - “Fider is a feedback portal for feature requests and suggestions”. Fider is primarily built using Go with the Gin framework. This application appeared to be a good candidate for this testing because it was multi-user, had multiple user roles (i.e., Visitor, Collaborator, and Administrator), allowed users to generate content and upload files, had both public and private functionality, and supported multiple authentication methods (e.g., Facebook, Google, GitHub, magic links).

Photoview (https://github.com/photoview/photoview) - “Photoview is a simple and user-friendly photo gallery that's made for photographers and aims to provide an easy and fast way to navigate directories, with thousands of high-resolution photos”. Photoview is primarily built using TypeScript with the NextJS framework. This application, similar to Fider, appeared to be a good candidate for this testing because it was multi-user, had multiple user roles, and allowed users to fetch and parse image files.

TEST ENVIRONMENT SETUP

Infrastructure

Both applications were configured and run on a fully updated Ubuntu 24.04.4 LTS, t2.medium AWS EC2 instance.

Target Deployments

The two targets were deployed using the following setup:

Fider Deployment

- Installation of the software was completed by following the project's documentation using the `docker-compose.yml` (see Appendix A)
- TLS was also configured by following the project's documentation
- The OAuth Provider setup was configured by following the *Google* section of this part of the project's documentation
- This Docker image was used: getfider/fider:v0.33.0, which ran Fider version: 0.33.0
- Two accounts (one "Administrator" account and one "Member" account) were created

Photoview Deployment

- Installation of the software was completed by following the project's documentation using the `docker-compose.yml` (see Appendix A)
- Standalone authentication
- TLS was provided using a simple nginx reverse proxy, which exposed HTTPS only (see Appendix A)
- This Docker image was used: photoview/photoview:2.4.0, which ran Photoview version: 2.4.0
- Two accounts (one admin and one user account) were created

Vendor Platform Configuration

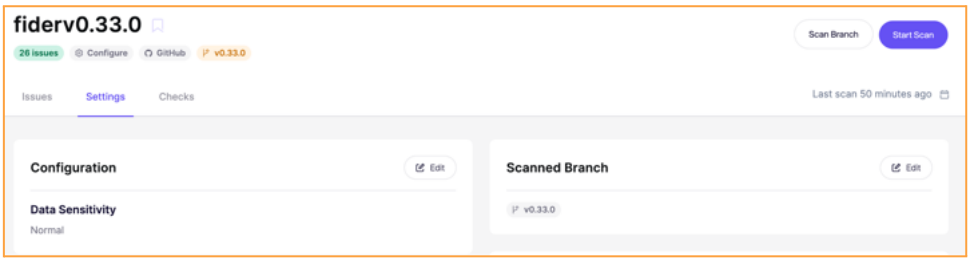
Overall, the setup process was largely as we expected. It involved creating an account, selecting the desired testing tier (cost, duration, and depth), and granting the platform access to the application's source code, together with URLs and credentials. We defined the testing scope (domains, URLs, and paths), verified domain ownership, and provided application user credentials for authenticated testing. Finally, we configured operational parameters such as testing windows, rate limiting, and any required custom headers.

Below are the step-by-step configurations needed to configure the tests in the manner we did. These are provided for repeatability and potentially for educational purposes as well.

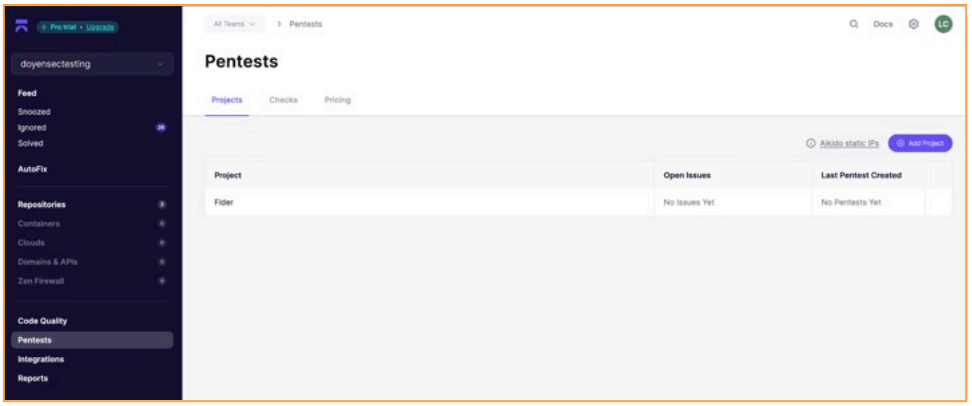
Aikido Fider Configuration

- Paired GitHub to fetch our fork of the repo

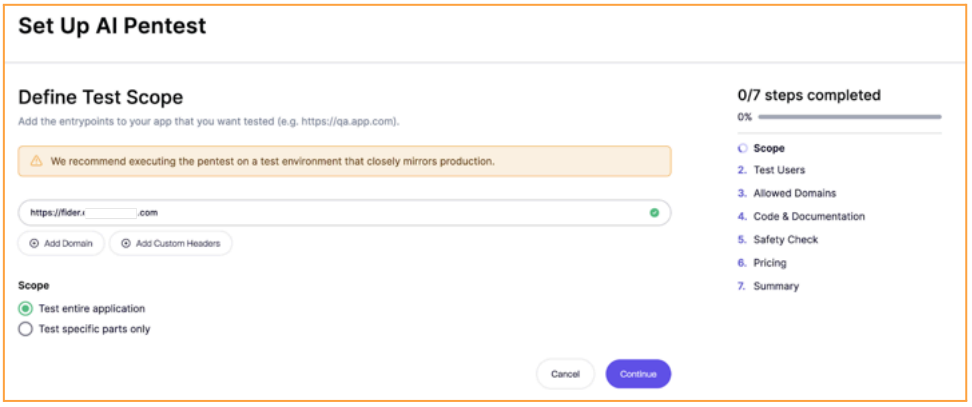
- Selected the specific scanner branch



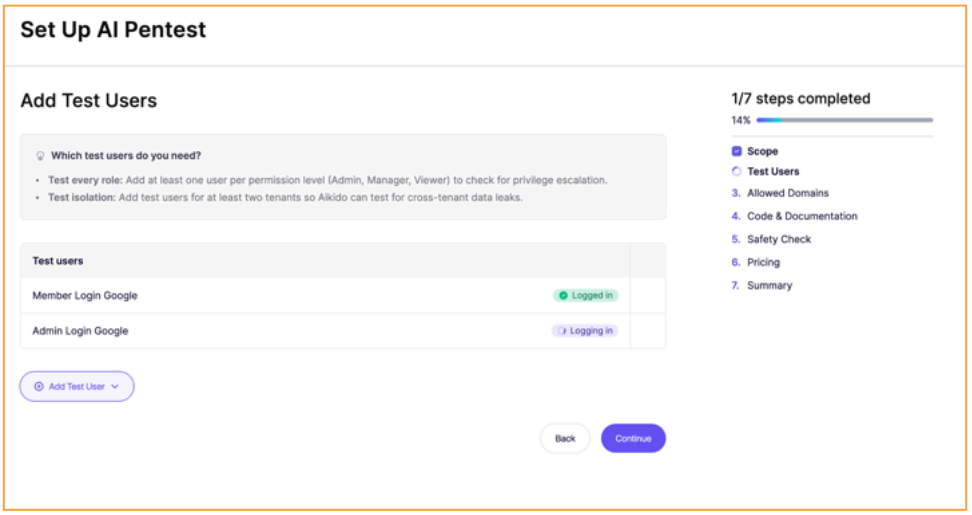
- In Pentest, created a new project



- Defined the “Test Scope” (URLs)



- Configured the testing users (one admin, one low-priv user), both of which were using the Google authentication option.

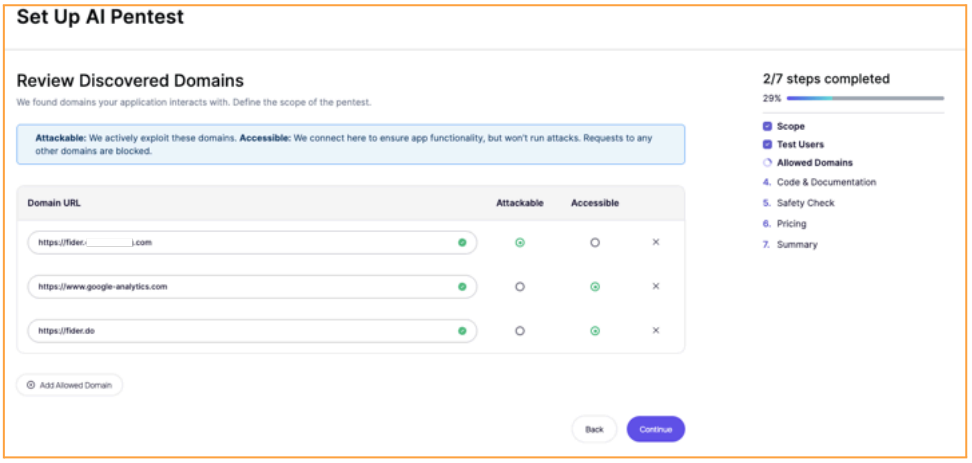


- This step required contacting Aikido's customer support team since the "Admin Login Google" failed to log in, despite being a copy of the other login (different user/pwd). It turned out to be Google blocking requests via the "Login Challenge", which was bypassed using this knowledgebase article.

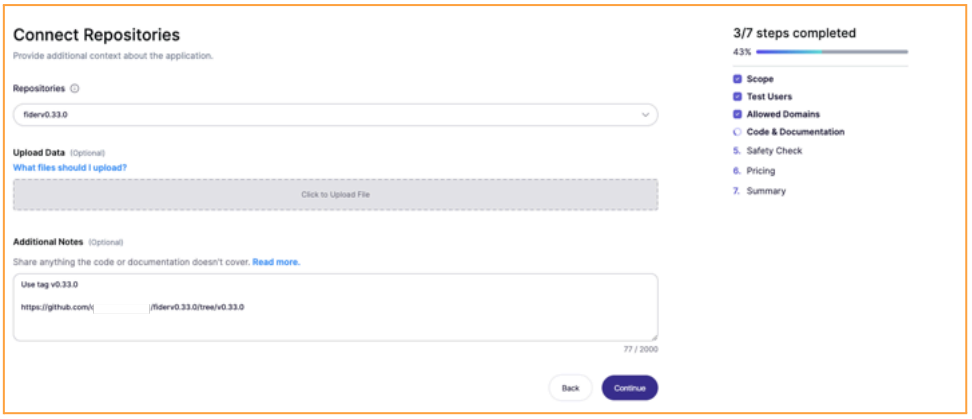
Our experience says that it is not uncommon for automated security tools to struggle with setting up authentication when it is more complex than the simplest of HTML forms. Working around this is sometimes very tedious. Therefore, we found it interesting that the Aikido configuration requires users to write a prompt to instruct the agent on how to authenticate using SSO. This seems to acknowledge those difficulties and demonstrates an evolution of the way these platforms are configured. This is the prompt we used to authenticate via Google:

```
1. Go to https://fider.<OMITTED>.com
2. Top right, click on "Sign In"
3. In the modal, select "Continue with Google". You will be redirected to accounts.google.com.
4. Enter email: testing@<OMITTED>.com and click "Next"
5. Enter password: <OMITTED> and click "Next"
Success criteria: You are redirected back to the app and the login was successful
```

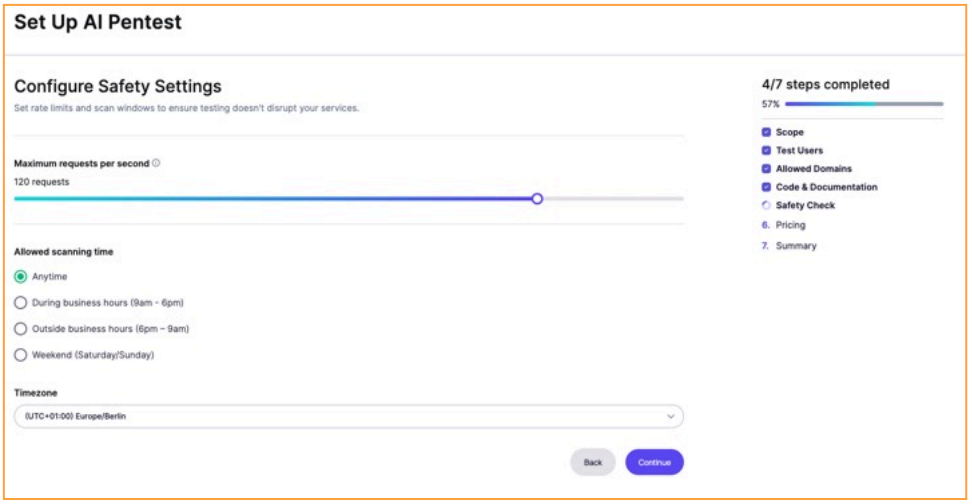
- Allowed domains



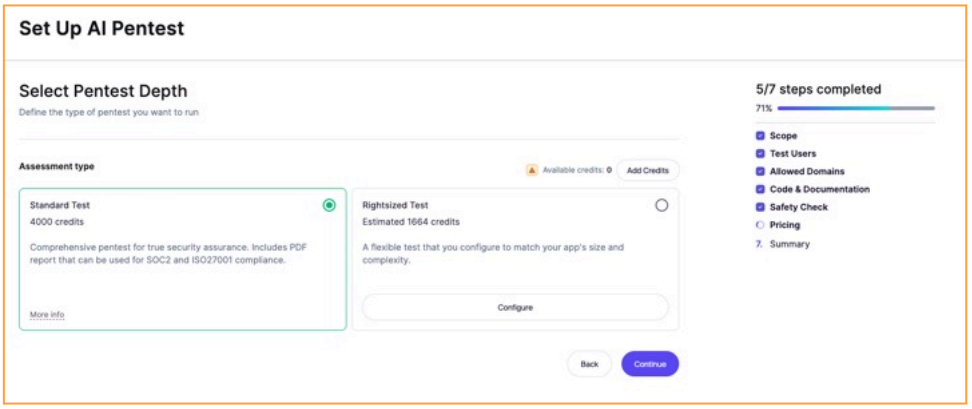
- Connected repositories



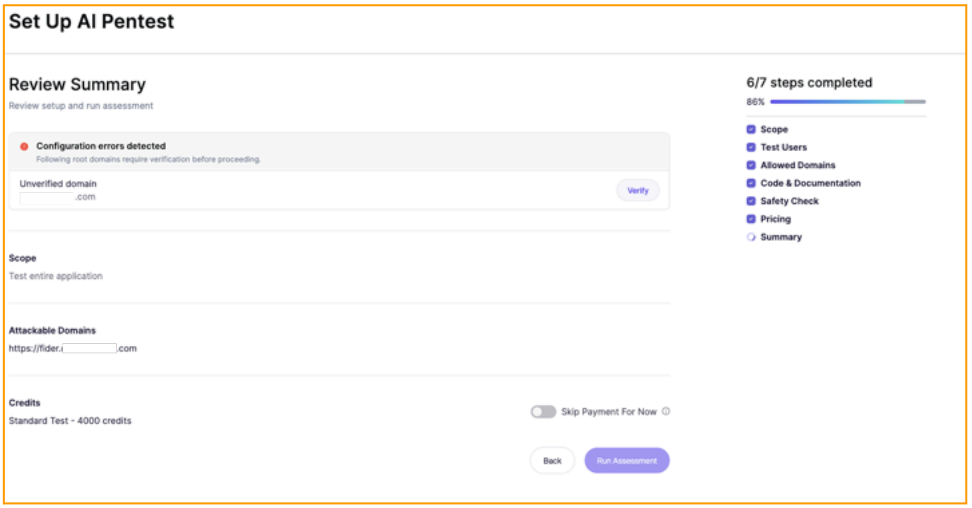
- Configured “Safety Settings”

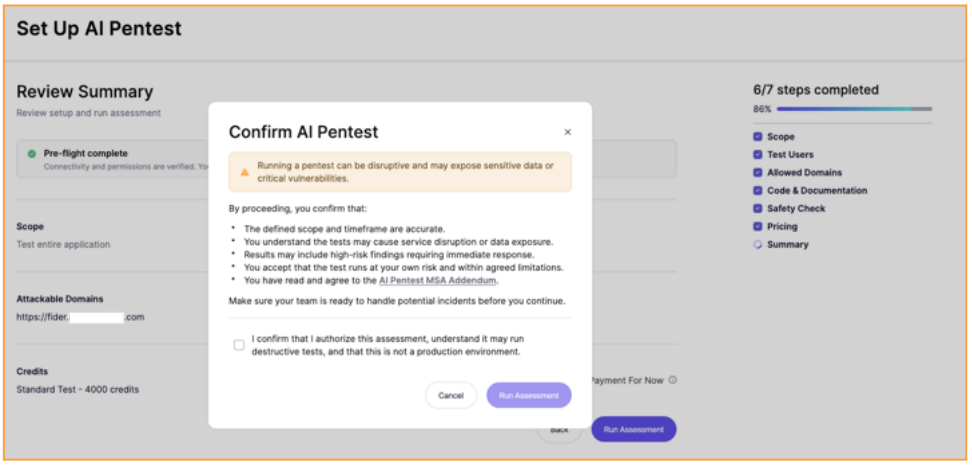


- Selected “Pentest Depth” - Standard Test (i.e., Pricing)



- Verified the domain





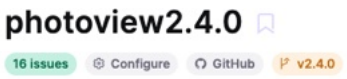
- The pentest completed without further interaction

From there, the rest of the process was completed via their web portal without human interaction, unless otherwise mentioned. **The testing was started and completed on March 23, 2026, and**

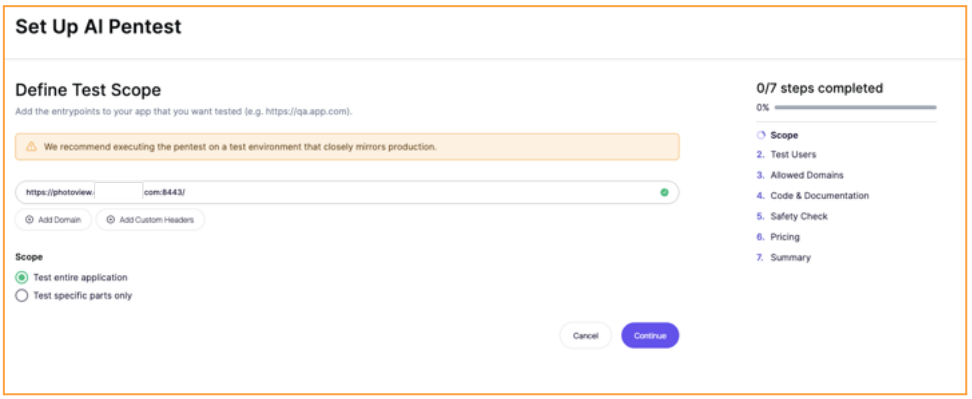
took approximately 8 hours and 40 minutes. The report was delivered immediately after the scan completion.

Aikido Photoview configuration

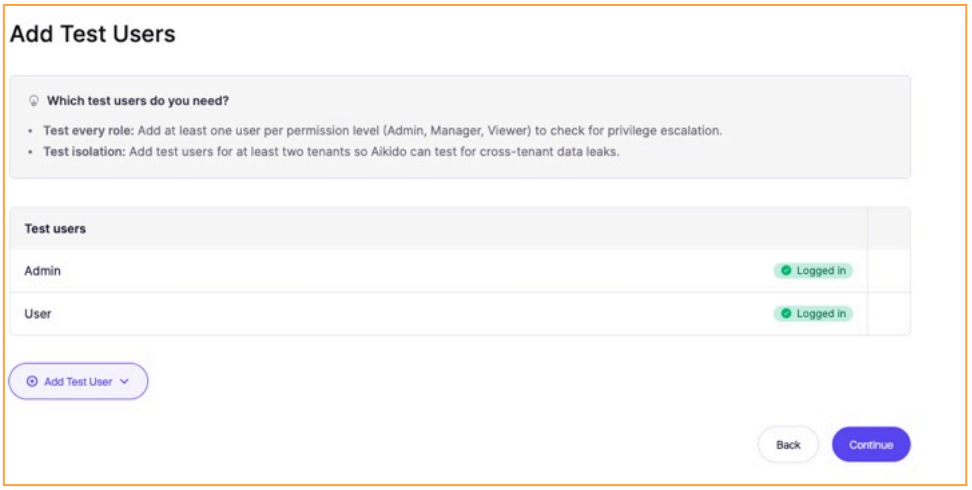
- Paired GitHub to fetch our fork of the repo
- Selected the specific scanner branch



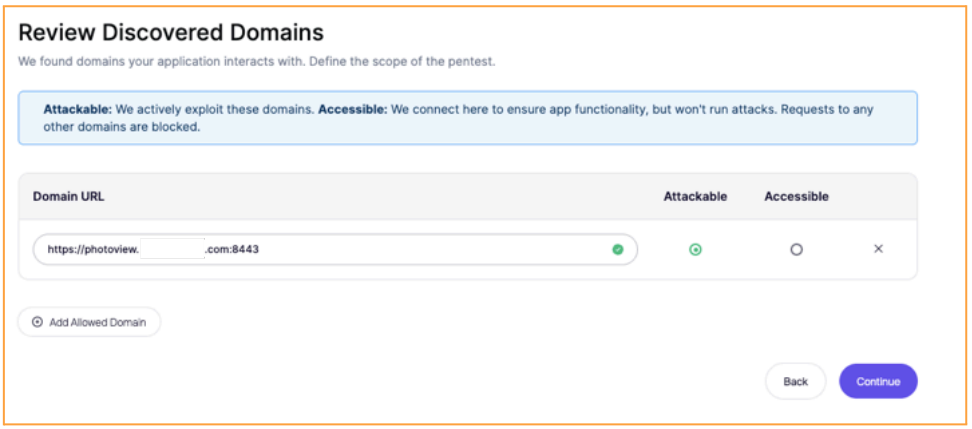
- Created a new pentest



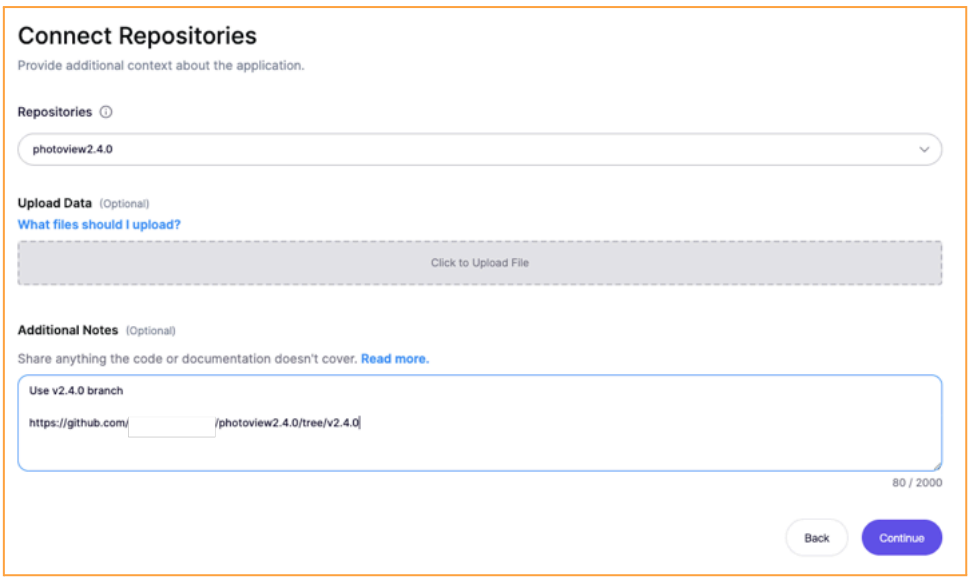
- Set up users (standalone credentials)



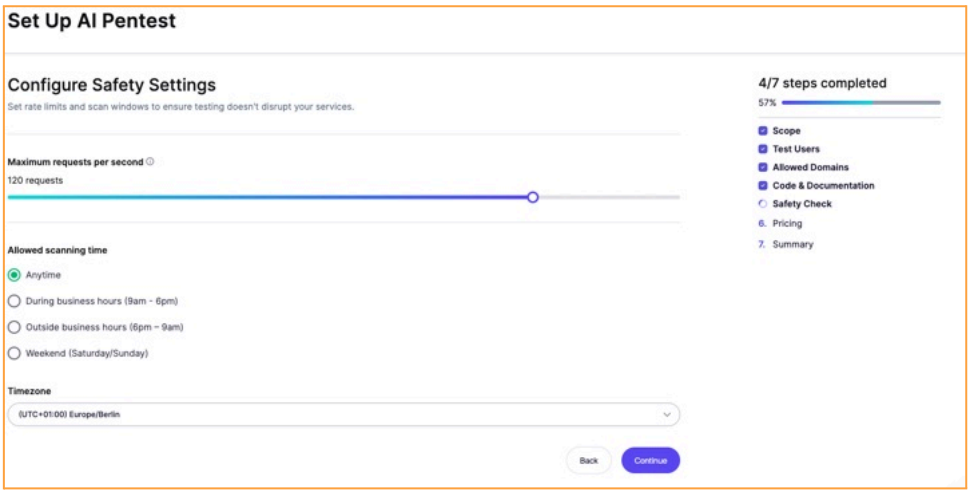
- Setup domains



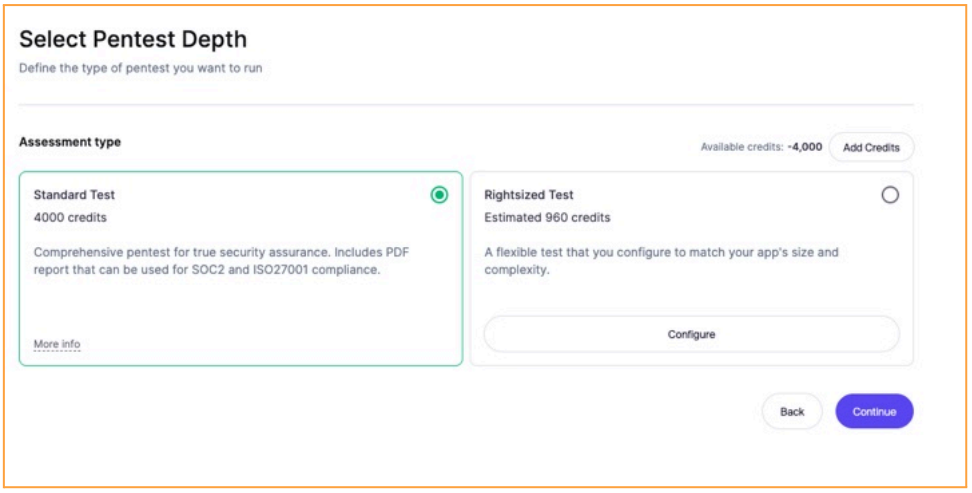
- Connected repositories



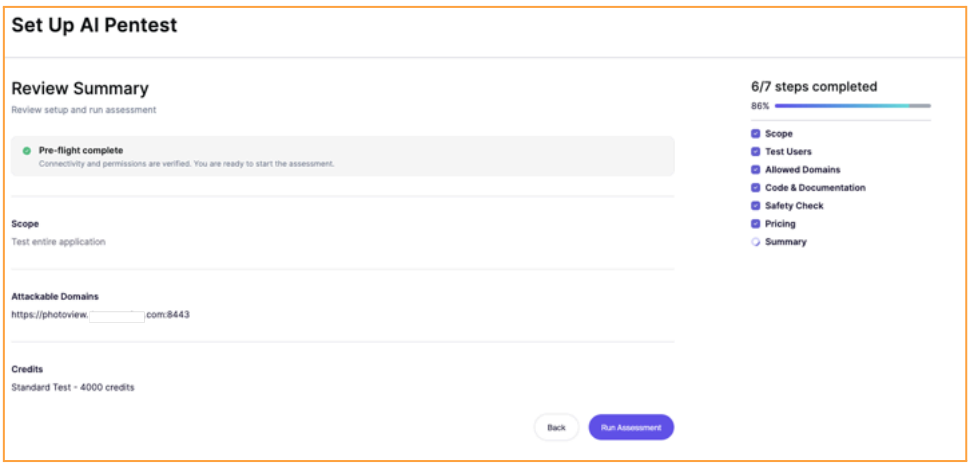
- Configured “Safety Settings”



- Selected “Pentest Depth” - Standard Test (4000 credits)



- Reviewed the summary and ran the assessment



As with the Aikido Fider testing, the rest of the process was completed via their web portal without interacting with them, unless otherwise mentioned. **The testing was also started and completed on March 23, 2026, and took less than 8 hours. The report was delivered immediately after the scan completion.**

XBOW Fider configuration

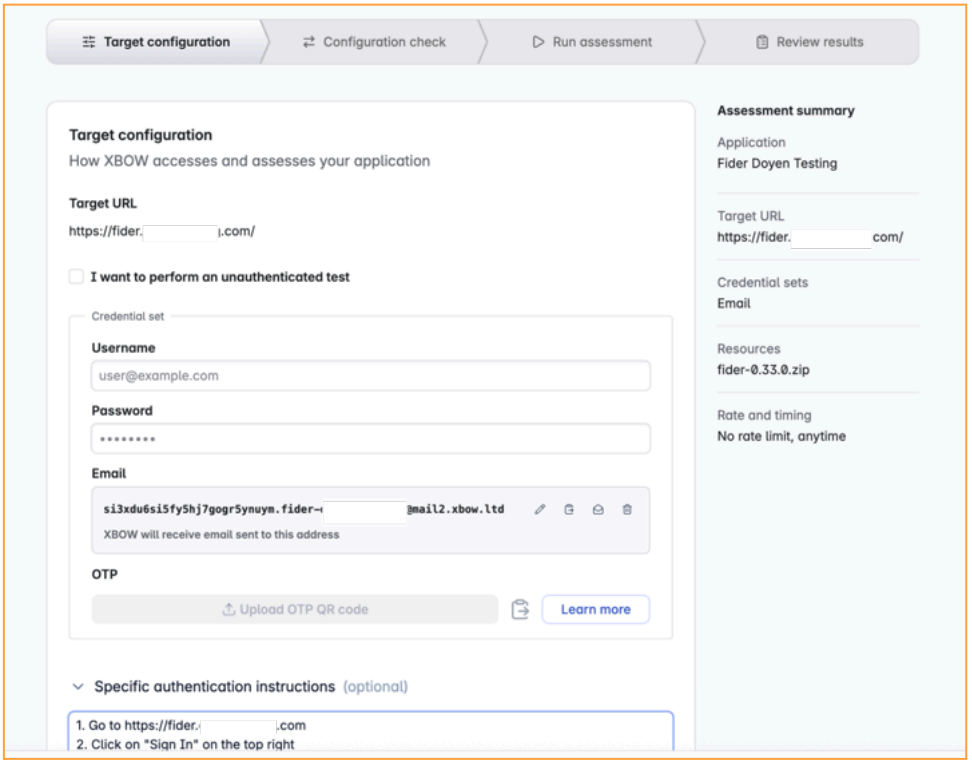
Similar to Aikido, XBOW provides a multi-step setup to define the source code and the testing environment for the target.

Given that **XBOW did not support Google Social Login (or other social platforms) at the time of testing**, we reverted to one-time email authentication. This secondary authentication mechanism was also supported by Fider, even though it was not our first choice for users.

The configuration for Fider was done according to XBOW's guidance in their XBOW Assessment Configuration Guidelines[1] document, dated January 2026, and guidance from their support team.

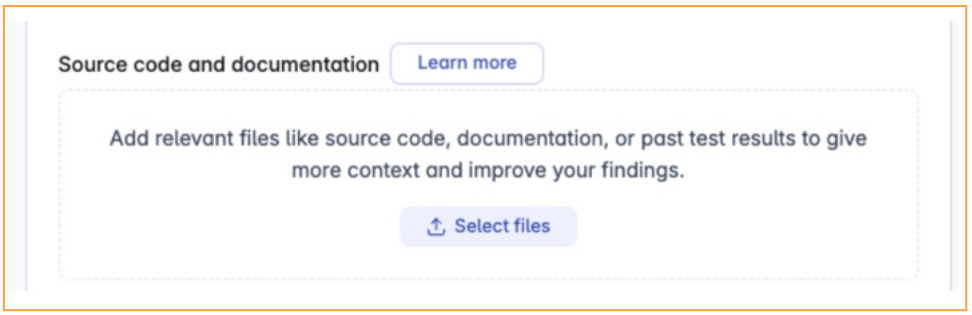
- *Target Configuration*

  Supplying the target URL and credentials (actual configuration shown).

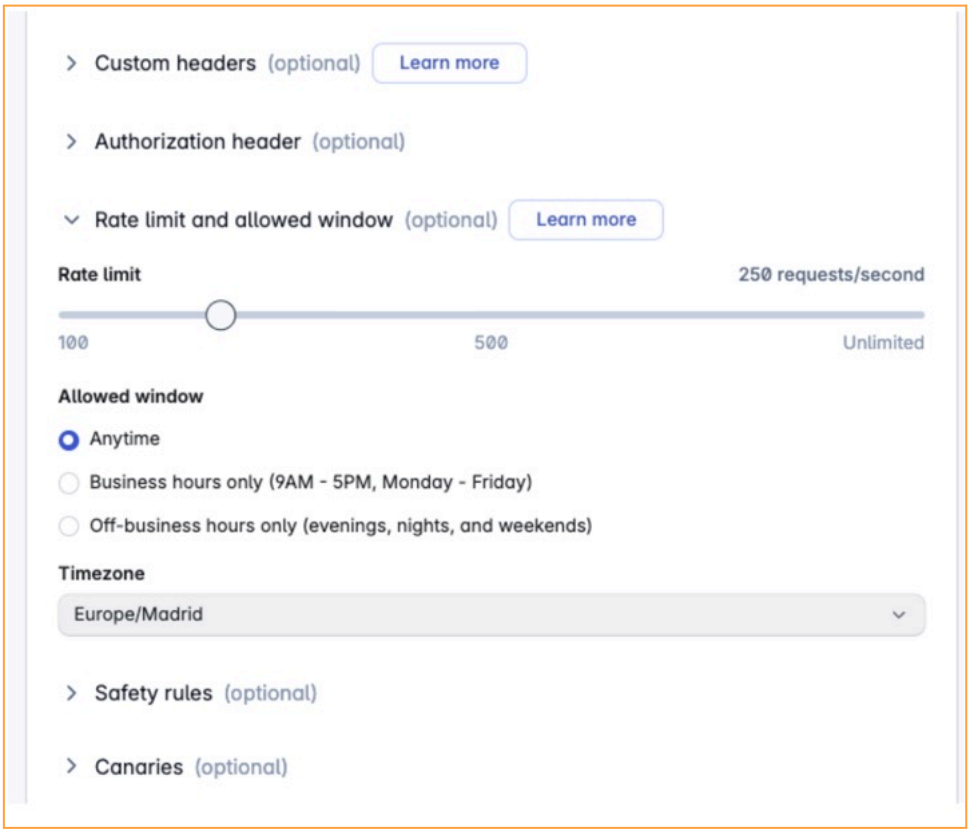


[1] SHA-256 hash of the configuration guidelines provided by the vendor: 9db9503e9be9de19e7949ab783da6cf0b63ecad86c9f296c101f434c8ea452a5

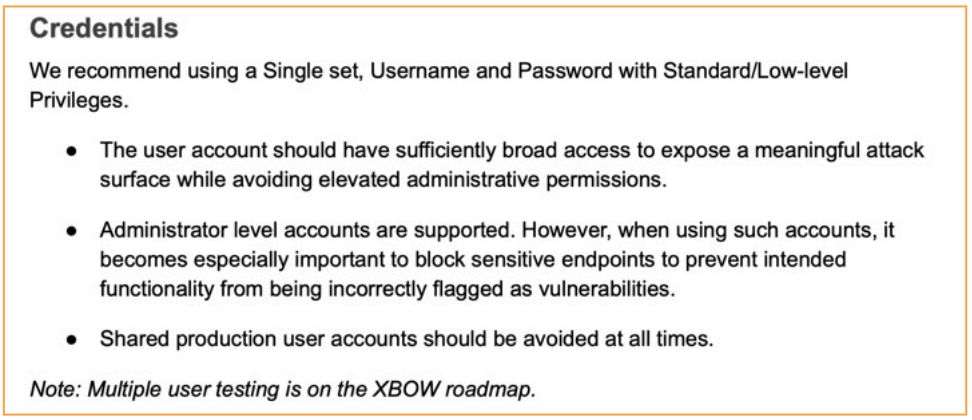
- Within the same window, we were also able to upload the source code, which the guide says will improve the results.



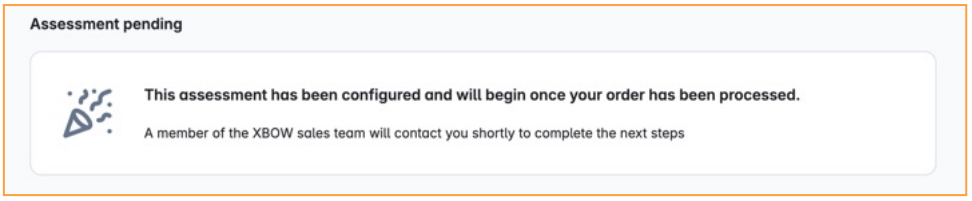
- We were also able to configure various testing parameters, such as time of day, rate limits, in-scope URLs, etc. We left these configurations at their default values.



- **XBOW does not currently support multi-user testing,** as documented in the "XBOW Assessment Configuration Guidelines". For this reason, a single set of credentials was provided. Despite the recommendation in the document, the customer support representative suggested using an admin user in order to expand the attack surface available for testing.



- Once all the necessary parameters were input, XBOW performed a confirmation of the settings (e.g., URL was available, credentials work)



- The overall flow was rather cumbersome as the initial setup required manual approval from the XBOW's sales team, and the contract signing was handled outside of the platform (standard DocuSign PDF signature). Finally, the scan was interrupted and restarted multiple times due to the following reasons (in order of appearance):
  - Disabled Authentication Mechanism
    - ➢ The scan modified the default authentication mechanism, which prevented further log in attempts. This was solved by blocking a specific endpoint.
  - Deleted Testing Account
    - ➢ The AI agent deleted the testing account, which prevented future log in attempts. This was solved by blocking a specific endpoint.
  - Loss of Connectivity (two occurrences)
    - ➢ XBOW testing infrastructure lost connectivity to the target. The EC2 instance was completely unresponsive. While we didn't pinpoint the exact root cause, this was likely caused by system resource exhaustion.
    - ➢ This issue was eventually resolved by upgrading the AWS instance type to t2.large and deleting the over 4000 entries from the database.
    - ➢ Additionally, our Mailgun subscription limit was increased to prevent quota issues, given that the application had sent over 4800 emails for one-time token logins and notifications.

Once all technical issues were resolved, the rest of the scan and report delivery were completed with no further delays. **The testing was started on April 10, 2026, and completed on April 16, 2026. The final report was delivered on April 21, 2026.**

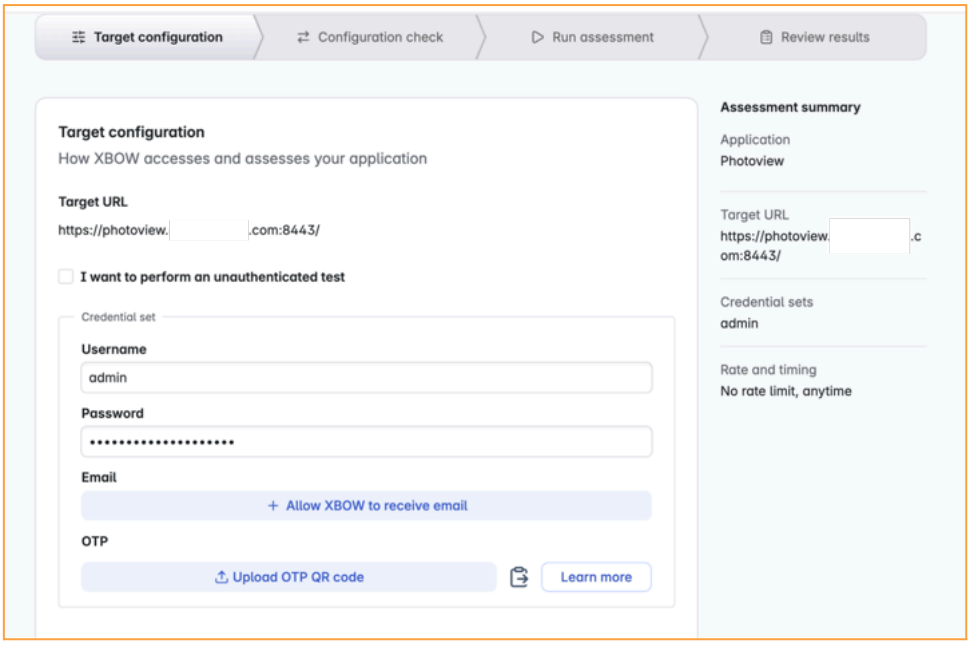
For completeness, the detailed timeline is presented here:

- April 8, 2026 - Submitted the “Start Your Pentest” online form, and notified the sales representative we previously spoke to
- April 8, 2026 - Received the order form via DocuSign
- April 9, 2026 - Received link to configure the scan
- April 10, 2026 - Scan was started
- April 16, 2026 - Scan completed after several pauses and restarts
- April 21, 2026 - Report delivered

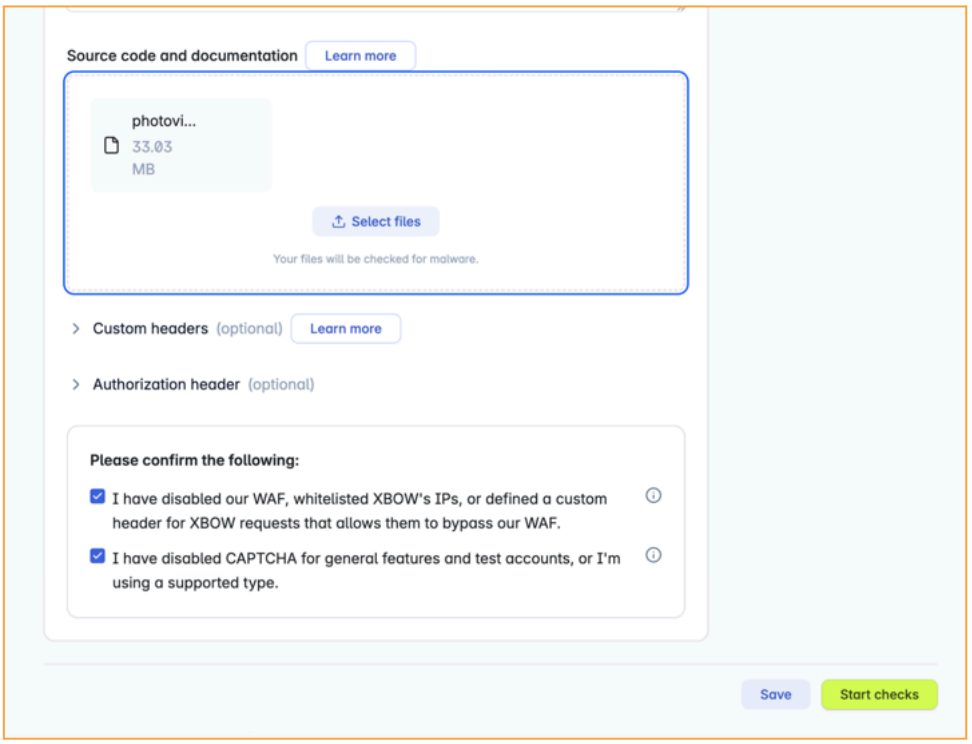
XBOW Photoview configuration

Just like in the Fider testing, the configuration for testing Photoview was done according to XBOW’s guidance in their “XBOW Assessment Configuration Guidelines” document, dated January 2026. Rather than repeat the example screenshots from the ones already displayed, we have just included some key images from the Photoview testing.

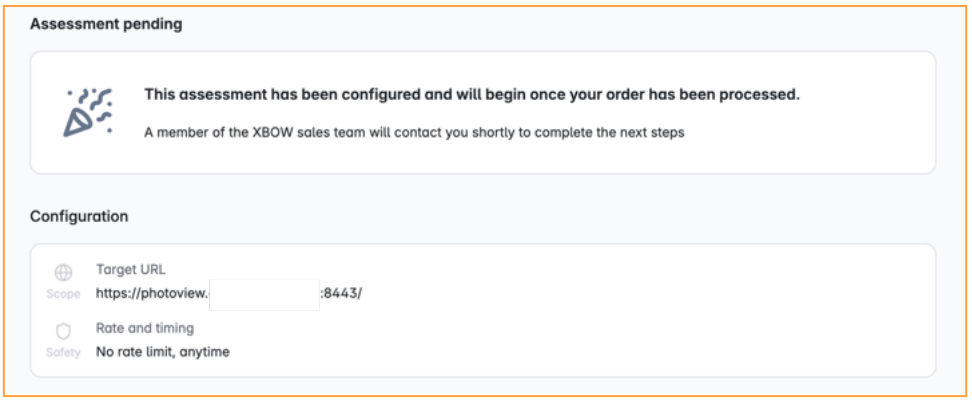
- This screenshot shows the initial configuration, including specifying credentials:



- This screenshot shows where users can upload source code and documentation along with some customizations:



- After the configuration, we received this completion screen, letting us know the order will begin once our order has been processed and that someone from their sales team would contact us to complete the process.



- Unlike the XBOW Fider testing, the scan was performed smoothly with no interruptions.

- The flow of communications for the testing:
  - March 31, 2026 - Submitted the “Start Your Pentest” online form
  - April 2, 2026 - A sales representative sent instructions and confirmed the target information
  - April 3, 2026 - Received the order form via Docusign
  - April 6, 2026 - Scan was started
  - April 8, 2026 - Test completed and report delivered

Operational Constraints

As mentioned above, XBOW didn't support social authentication. Further, XBOW didn't support specifying multiple users for a test, so we could only configure it for one account. The XBOW Support team recommended using a single admin account in order to provide better coverage. Due to the use of an admin account, we were forced to exclude certain URLs for the Fider application, whose inclusion would have caused errors in the application. It is therefore possible that some authentication-related vulnerabilities could have been missed as a result, but we did not feel there was a workaround for this, and mentioning this would be noteworthy for others.

Time limits for scans

We did not set any time limits for scans, allowing them to complete per the platform's own internal processes, within the allocated budgets.

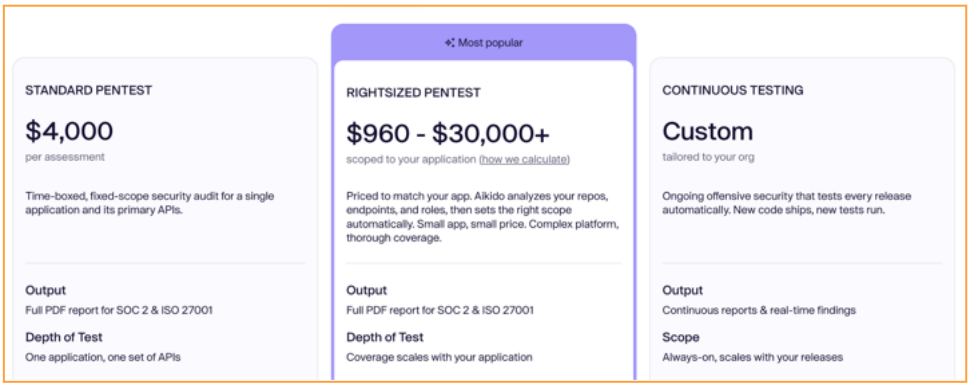
Default vs. Tuned Configurations

We did not configure any tuning parameters (e.g., rate limits) on the testing beyond the defaults. It is possible that modifying some of these could have resulted in a better experience and not caused outages in our applications and/or the bulk email service we used.

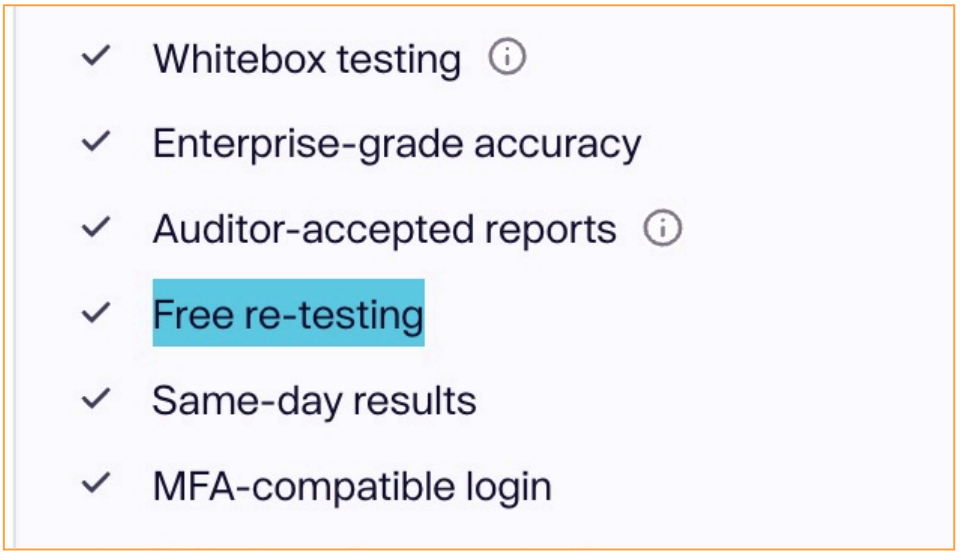
COST CONSIDERATIONS

License Options

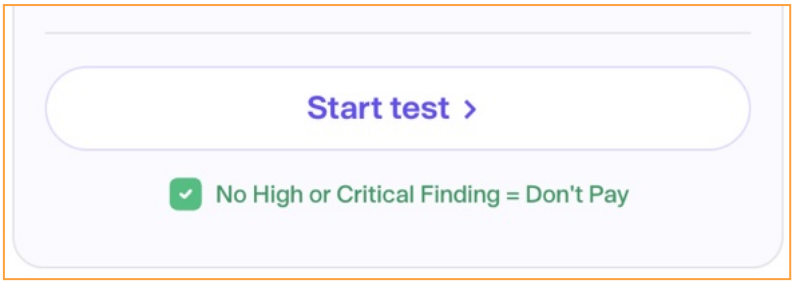
Currently, Aikido's penetration testing options include a "Standard Pentest" for $4,000 USD, a "Rightsized Pentest" for $960-$30,000+ USD, and a customizable option. Per our agreement with Aikido, we opted to use the "Standard" version in order to be of comparable cost with the XBOW's equivalent. It is fair to note that Aikido provided a credit estimator to tune the scan to the specific size/complexity of the target. Independent of the result of the estimator, we opted for using the 4000 credits ($4000 USD) option.



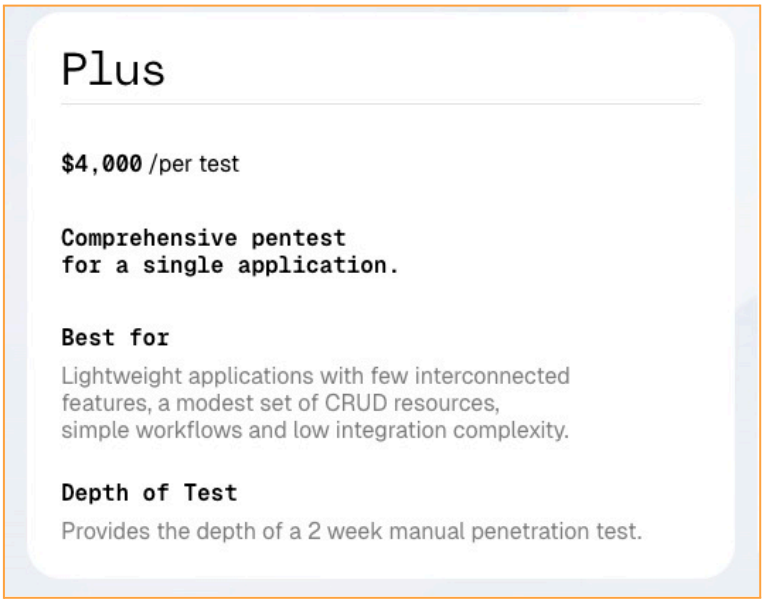
All of the testing levels appear to offer free re-testing. Aikido informed us that this is valid for an unlimited number of re-tests within 90 days.



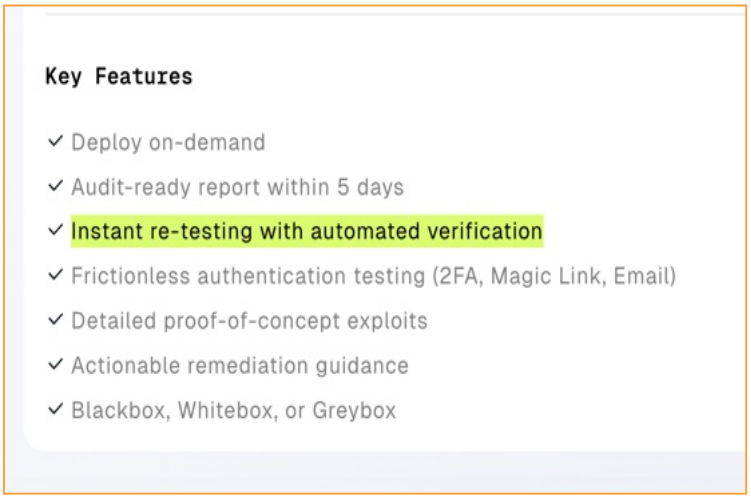
It appears they also offer a guarantee if there isn't at least one "Critical or High Finding". Given that both scans resulted in High or Critical vulnerabilities, we did not verify this.



XBOW's penetration testing options included $4000 USD - "Plus", $8000 USD - "Premium", and custom-priced "Enterprise" versions.



XBOW appears to offer a one-time retest within 30 days of the completion of the audit at no additional cost.



Since it's difficult to understand and compare the exact details differentiating the offerings, we equated them based on price, as we believe most organizations would. We therefore assumed XBOW's "Plus" option corresponded roughly with Aikido's "Standard" version.

FINDINGS EVALUATION

Manual Validation

Upon receipt of the results from the platforms, we performed verification via **manual code reviews**. During this process, we used the descriptions provided by the platforms to accelerate the review, rather than starting from scratch. This also allowed us to evaluate how well the platforms did at creating useful descriptions. Whenever possible, **we also performed dynamic testing to confirm the vulnerability was real and exploitable.**

The tables below show how much time was spent on validating the findings (including those found to be false positives). It's important to note that each platform's results were evaluated by a different researcher, meaning the times are not copied from one platform to another for any overlapping findings.

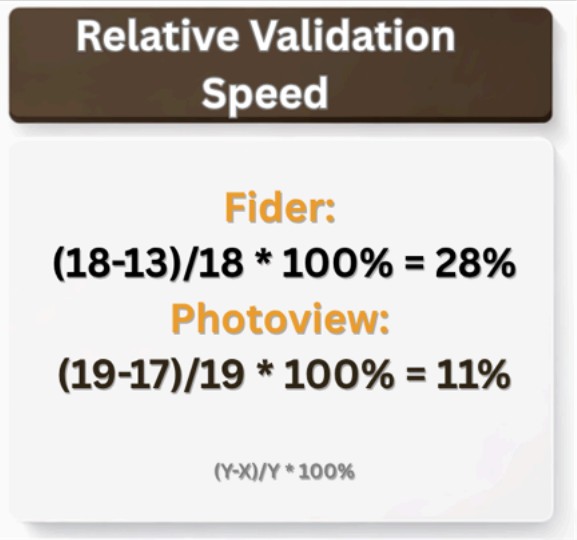
Aikido Validation Times

App	Average Time (mins)	Total Time (mins)
Fider	18	348
Photoview	19	622

XBOW Validation Times

App	Average Time (mins)	Total Time (mins)
Fider	13	426
Photoview	17	121

Since we assume the researchers validating the findings were roughly equally capable, it's then possible that XBOW **may** have made validation easier in some way (e.g., clearer descriptions, better examples). Per the calculations below, the validation time for XBOW was 11-28% less.



This result was surprising to us, because the researcher validating the XBOW findings actually reported that the XBOW reproduction steps sometimes included scripts that were not usable because they were not properly fit/framed in the PDF.

As with all the findings, caveats apply. Things like the similarity of the findings (e.g., multiple of the same type) or simplicity of the findings could make triaging faster, without necessarily indicating the value provided by the platform. One example that illustrates this point is the redundancy of XBOW-FIDER-2 and XBOW-FIDER-5. They were reported as two different classes of vulnerability, both equally valid, but with the same root cause. Because of this, the first one took sixty-two (62) minutes to validate, whereas the second only took fifteen (15) minutes due to the similarity. To demonstrate this, just removing that one effectively duplicate finding would result in adjusting the average time spent for Fider validation to be: $(426 - 15)/(24 - 1) = 411/23 = 17.9$, which is the same average time Aikido took. To be fair, we would need to perform this same type of additional analysis across all the results, for both platforms, to get the "fairest" final values.

Classification Criteria

We considered vulnerabilities as true positives (TP) as long as they were real/exploitable. If it were real, but not impactful, we would mark it as *informational*.

Looking at the data, we ended up changing (mostly lowering) severities more than disputing the validity of reported findings. Determining the causes of this potential tendency for these platforms to overestimate impacts is beyond the scope of this research.

Accurately assessing impact depends heavily on understanding an application's context. What qualifies as "sensitive" data or functionality varies from one system to another, making it difficult

to determine severity in isolation. As a result, the very same vulnerability may receive vastly different scores once the full context of the application is understood.

Disclosure

Doyensec spent two additional working days preparing a report of the manually confirmed vulnerabilities for both target OSS projects. The Fider vulnerabilities were reported to the maintainer on 04/30/2026, while the Photoview vulnerabilities were reported to the maintainer on 05/01/2026. For this reason, this study is published with a redacted spreadsheet of the results, given that some issues have not yet been fixed by the maintainers. Future versions of the spreadsheet might be released.

RESULTS

Overview of Findings

Both of the platforms found numerous classes of vulnerabilities, provided reasonable reports, and overall returned a low percentage of false positives, but were less accurate in terms of the severity. As professional security testers, we recognize that both platforms provide a solid baseline for security scanning, offering a better return on investment compared to traditional enterprise SAST/DAST solutions.

During this research, it was necessary to track individual reported vulnerabilities, particularly for comparisons. Since XBOW did not provide unique identifiers for each vulnerability, we created our own IDs by simply concatenating the platform name, followed by the project being scanned, plus an incrementing integer (e.g., XBOW-FIDER-1). A similar approach was taken for Aikido. The list of all the findings and their corresponding IDs will be included in an archive file released with this paper.

Aikido Results

Fider Test Results

In total, nineteen vulnerabilities were reported by the Aikido platform. These included a wide range of vulnerabilities, including mass assignment, missing rate limiting, lack of data sanitization, race conditions, information leakage, open redirection, path traversal, missing/misconfigured security headers, and authentication and authorization vulnerabilities.

Per our team's review of the validity of the findings, we determined two of them were false positives. This included a *high-severity* "Host-header poisoning in sign-in email flow allows magic-link hijacking and account takeover", where we weren't able to reproduce the vulnerability dynamically, and after looking at the source code, did not find anywhere that the Host header was used to construct the URL sent via the email.

The second false positive was a "Race condition in /_api/signin/verify allows one-time OTP to be redeemed multiple times". In this one, we weren't able to reproduce the vulnerability, and felt even if we were, there would have been no impact due to a race condition in the specified location.

Based on the above, we would summarize the results numerically as:

- 89% (17/19) of the reported vulnerabilities were true positives
- 11% (2/19) of the reported vulnerabilities were false positives

Summarized notes of interest from the review are shown below. All others should be assumed to be acceptable and not noteworthy:

- AIKIDO-FIDER-1: The exploit is not fully reliable.

- AIKIDO-FIDER-2: The reproduction steps don't fully prove that the missing rate limit is exploitable.

- AIKIDO-FIDER-11: The provided payload needed adjustments (escaping was missing), and the description was lacking some of the potential impacts possible.

Per the Aikdo platform's internal severity scoring system, the severities ranged from *low* to *high*. Our team would have scored the true positives between *low* and *critical*.

Of the remaining 17 verified true positives, according to our scoring:

- 76% (13/17) of the reported vulnerabilities were assigned the correct severity
- 24% (4/17) of the reported vulnerabilities were assigned an incorrect severity

Per our assessment, for the Fider application, the Aikido platform overstated the severity of three vulnerabilities and understated one. The severities assigned were within one level of our classifications, so they weren't significantly miscategorized.

The adjusted severities are detailed in the table below:

Adjusted Severities

Finding ID	Finding	Original Severity	Adjusted Severity	Reason
AIKIDO-FIDER-1	Mass assignment in sign-in binder enables unauthenticated account takeover via controlled OTP/link key	High	Critical	The severity was raised since this is a pre-auth account takeover which we usually report as Critical
AIKIDO-FIDER-10	50881bf9cc73ebf5e7aaa8e22569f47700dd7a6c	Medium	Low	The severity has been downgraded since only the user ID can be extracted when exploiting this vulnerability
AIKIDO-FIDER-12	cdcc3e46a23b6bac338d6f78d3fab4365279ebcc	Medium	Low	The severity has been downgraded since the actual impact is an [redacted]
AIKIDO-FIDER-13	c3c183c10c5b0036394db34b47d564ddc807f8df	Medium	Low	The severity has been downgraded since the actual impact is an [redacted]

Assigning severities can be somewhat subjective, so some variance is not surprising. We take no stance on whether overstating or understating severity is preferred, since both have consequences.

Photoview Test Results

In total, thirty-two vulnerabilities were reported by the Aikido platform. These included a wide range of vulnerabilities, including SQL injection, broken authorization (e.g., IDOR), information leakage (e.g., enumeration, verbose errors), broken session management, missing rate limiting, GraphQL configuration-based vulnerabilities, missing/incorrect cookie flags, insecure TLS configuration, and authentication and authorization vulnerabilities.

Per our team's review of the validity of the findings, we determined thirty-two were true positives, and there were no false positives.

Based on the above, we would summarize the results numerically as:

- 100% (32/32) of the reported vulnerabilities were true positives
- There were no false positives reported

Summarized notes of interest from the review are shown below. All others should be assumed to be acceptable and not noteworthy:

- AIKIDO-PHOTOVIEW-10: The reproduction steps don't make any sense in context.
- AIKIDO-PHOTOVIEW-13: The reproduction steps don't work.
- AIKIDO-PHOTOVIEW-31: The finding description is really vague.

Per the Aikido platform's internal severity scoring system, these ranged from *low* to *critical* severity. Based on our review, our team would have scored them between *informational* and *critical* severity. We felt that one *high* severity should be classified as *medium* severity, two *medium* should be considered *low* severity, three *medium*, and five *low* severity vulnerabilities should have been assigned an *informational* severity level.

According to our scoring:

- 66% (21/32) of the reported vulnerabilities were assigned the correct severity
- 34% (11/32) of the reported vulnerabilities were assigned an incorrect severity

With respect to the Photoview results, the Aikido platform overstated the severity of all eleven vulnerabilities. The severities assigned were within one or two levels of our classification. The general trend seemed to be that the platform didn't fully understand the sensitivity/context of the data involved with the reported vulnerabilities.

The adjusted severities are detailed in the table below:

Adjusted Severities

Finding ID	Finding	Original Severity	Adjusted Severity	Reason
AIKIDO-PHOTOVIEW-5	5ab2c0904545d0f9bd29ec3206a9d2ca86cb7247	High	Medium	The leaked info is not sensitive: Only image path, title, and id are leaked; This vulnerability grants no further access.
AIKIDO-PHOTOVIEW-8	1689e6dd739d68c3e133fefa3af39f7abe00b6ce	Medium	Low	The severity has been lowered since there is no clear impact besides some [redacted] added to the list of

Adjusted Severities

				[redacted]
AIKIDO-PHOTOVIEW-11	4c52b1205a5a09f1baf17af3f7c8846c54b2289a	Medium	Low	The severity has been lowered since the leaked information is minimal.
AIKIDO-PHOTOVIEW-18	026dbc268e77c0f6cfc617fdb7d9b7cd189179cd	Medium	Informational	The severity has been lowered since the leaked information is minimal.
AIKIDO-PHOTOVIEW-19	7307b8904b8084b626492317bb896edbb85c2fe5	Medium	Informational	The severity has been lowered since the leaked information is minimal.
AIKIDO-PHOTOVIEW-20	ad29c31e10d7dde2e34e77f89ec4387c4e2981d4	Medium	Informational	The severity has been lowered since the leaked information is minimal.
AIKIDO-PHOTOVIEW-21	8286bb5a8c0fdd5794eff5f3561f57d956d001d9	Low	Informational	The severity has been lowered since the leaked information is minimal.
AIKIDO-PHOTOVIEW-22	83efec4f852578323681fbebe56ebcc7be9e66e1	Low	Informational	While this is a weak security posture, it does not grant an attacker any additional access.
AIKIDO-PHOTOVIEW-25	13bc1bb6ba4c19d768e76a3aeb24ae078924f855	Low	Informational	While this is a weak security posture, it does not grant an attacker any additional access.
AIKIDO-PHOTOVIEW-26	77838d007f73d85a63367c7c4b5ba384fd0fb2e0	Low	Informational	The severity has been lowered because the disclosed information is limited to identifying the [redacted] in use.
AIKIDO-PHOTOVIEW-29	c90d4c3da3291e22d1100a838d011f52040e0279	Low	Informational	The severity has been lowered because the vulnerability is limited to [redacted]. While it reveals the application's [redacted] it does not allow an attacker to [redacted].

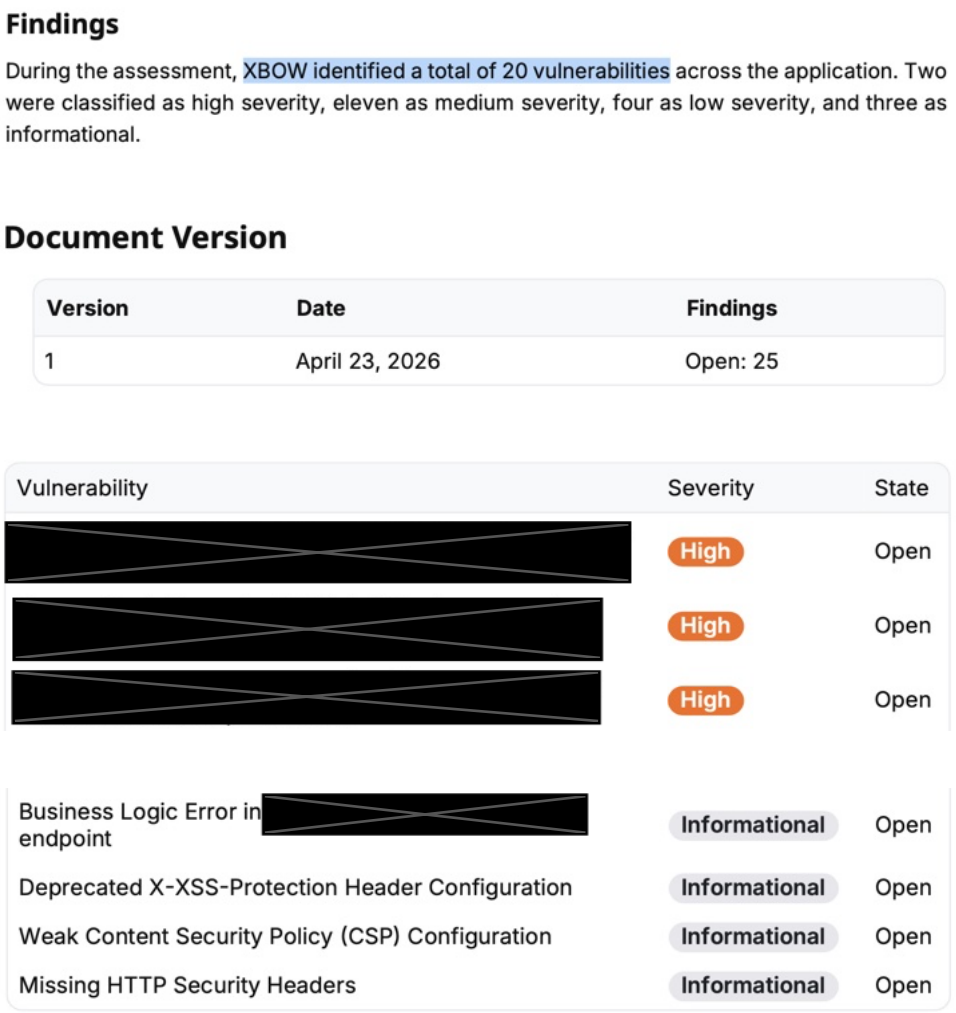
Again, assigning severities can be somewhat subjective, so some variance is not surprising. We take no stance on whether overstating or understating severity is preferred, since both have consequences.

XBOW Results

Fider Test Results

Before diving into the details, we wanted to highlight that the XBOW report was not internally consistent, as shown in the following excerpt images. Specifically, the number of vulnerabilities listed in the *Findings* section's text was lower than what was in the *Findings* column of the *Document Version* section's *Version* table. Then, later in the *Findings* section, the table lists twenty-five vulnerabilities. While investigating these discrepancies, we found one additional *High*, three additional *Medium*, and one additional *Informational* vulnerability compared to the description. **It's unclear whether this was a result of a bug in their report generation (e.g., miscounting) or due to manual modifications after the report was produced.** If the case is that additional vulnerabilities were added, it's unclear specifically why or how (i.e., via AI or human).

Additionally, the XBOW platform opted not to include any type of finding IDs. This is a deviation from the industry norm and makes navigating the document more difficult. The IDs used in this paper were created by us.



Findings

During the assessment, XBOW identified a total of 20 vulnerabilities across the application. Two were classified as high severity, eleven as medium severity, four as low severity, and three as informational.

Document Version

Version	Date	Findings
1	April 23, 2026	Open: 25

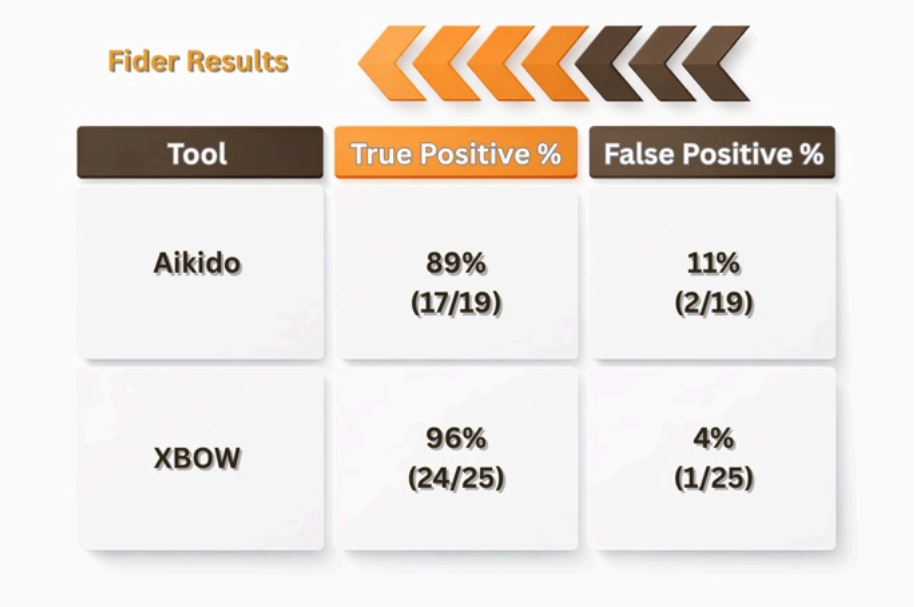
Vulnerability	Severity	State
[redacted]	High	Open
[redacted]	High	Open
[redacted]	High	Open
Business Logic Error in [redacted] endpoint	Informational	Open
Deprecated X-XSS-Protection Header Configuration	Informational	Open
Weak Content Security Policy (CSP) Configuration	Informational	Open
Missing HTTP Security Headers	Informational	Open

Moving on to the results, there were twenty-five vulnerabilities in total reported by the XBOW platform. These included XSS, RCE, Path Traversal, SSTI, SSRF, Open Redirection, a logic error, various Information Disclosures, and missing/incorrect security headers.

Per our team's review of the validity of the findings, we determined twenty-four were true positives, and one was a false positive.

Based on the above, we would summarize the results numerically as:

- 96% (24/25) of the reported vulnerabilities were true positives
- 4% (1/25) of the reported vulnerabilities were false positives



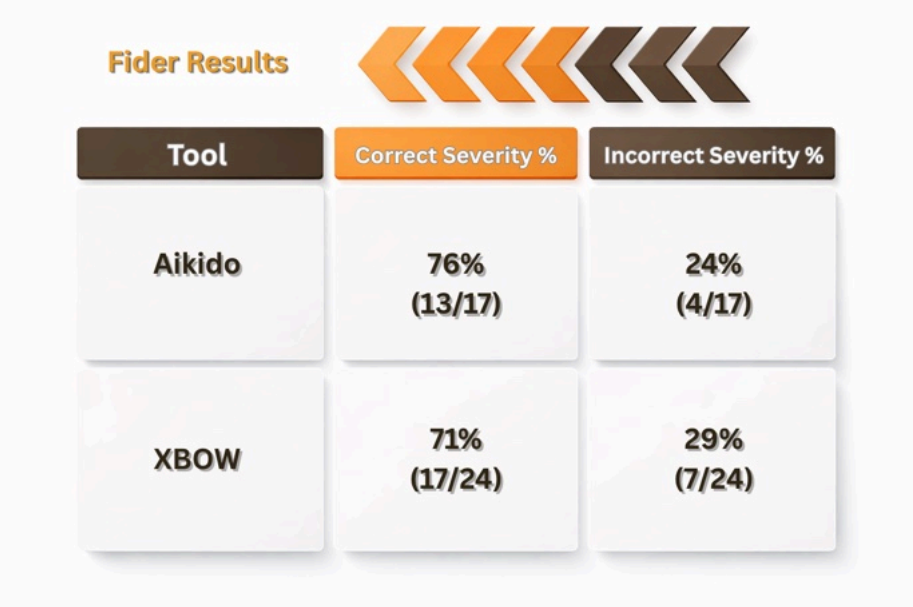
Summarized notes of interest from the review are shown below. All others should be assumed to be acceptable and not noteworthy:

- XBOW-FIDER-16: The reproduction steps are not complete.
- XBOW-FIDER-18: The reproduction steps are not fully usable.

Per the XBOW platform's internal severity scoring system, these ranged from *informational* to *high* severity. Based on our review, our team would have scored them between *informational* and *high* severity as well. However, we felt five *medium* severity vulnerabilities should be classified as *low*, and two more should have been classified as *informational* severity vulnerabilities.

According to our scoring (after removing the false positive):

- 71% (17/24) of the reported vulnerabilities were assigned the correct severity
- 29% (7/24) of the reported vulnerabilities were assigned the incorrect severity



With respect to the Fider results, the XBOW platform demonstrated a tendency to overstate the severity of vulnerabilities, where we disagreed. The severities assigned were typically within one level of our classification, except for two - one where it seemed not to understand the context, and another where the reported vulnerability resulted in no impact.

The reasoning for each is given below:

Adjusted Severities

Finding ID	Finding	Original Severity	Adjusted Severity	Reason
XBOW-FIDER-4	7181ae202577718108d852fb3fb7454a0dcc9023	Medium	Low	Severity lowered. This vulnerability can be used as an oracle to know [redacted] but it serves no further purpose.
XBOW-FIDER-7	7d10ebc04cf9f42ca075a63912ea219f628d4c1a	Medium	Informational	[redacted] le by using [redacted] hence there is no actual impact.
XBOW-FIDER-11	0f0836ee3debd8cec96274014b140baeaf749c14	Medium	Informational	There is no actual impact; a malicious user is only able to [redacted] without providing additional data/modifying any resource.
XBOW-FIDER-13	b34017fd2115354baf7887d1b935b62a1d6d1c32	Medium	Low	The severity was lowered because the [redacted] and requires administrative privileges to exploit. Additionally, the presence [redacted] further limits the potential impact.
XBOW-FIDER-15	7b1677edc7d2d69e4d803ad4ac17163ec6330e3d	Medium	Low	This finding severity was lowered because it can be used by the admin-only.
XBOW-FIDER-16	f619ededb555397c55189b164d49b6f92f4f03e3	Medium	Low	The severity was lowered because the vulnerability is [redacted] and requires administrative privileges to exploit. Additionally, the now default [redacted]

Adjusted Severities

				further limits the potential impact [redacted] environments.
XBOW-FIDER-17	29feda33fca3014a3099264dc9ebd1784d1d0c98	Medium	Low	The severity has been downgraded since the actual impact is an [redacted]

Again, assigning severities can be somewhat subjective, so some variance is not surprising. We take no stance on whether overstating or understating severity is preferred, since both have consequences.

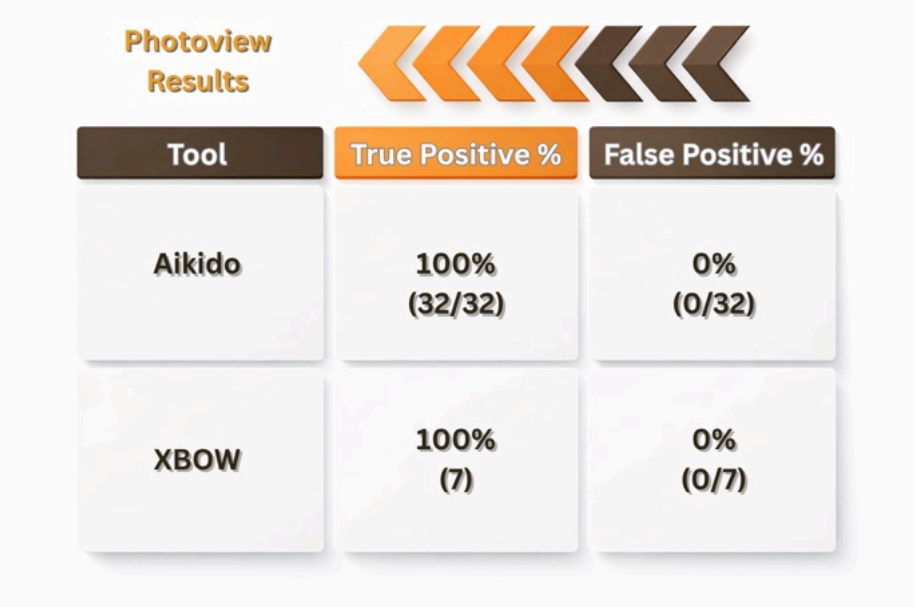
Photoview Test Results

In total, seven vulnerabilities were reported by the XBOW platform. These included a wide range of vulnerabilities, including SQL injection, Server Side Request Forgery (SSRF), information leakage (e.g., directory listing, verbose errors), broken authorization (e.g., IDOR), and missing/incorrect security headers.

Per our team's review of the validity of the findings, we determined all seven were true positives, and there were no false positives.

Based on the above, we would summarize the results numerically as:

- 100% (7/7) of the reported vulnerabilities were true positives
- There were no false positives reported



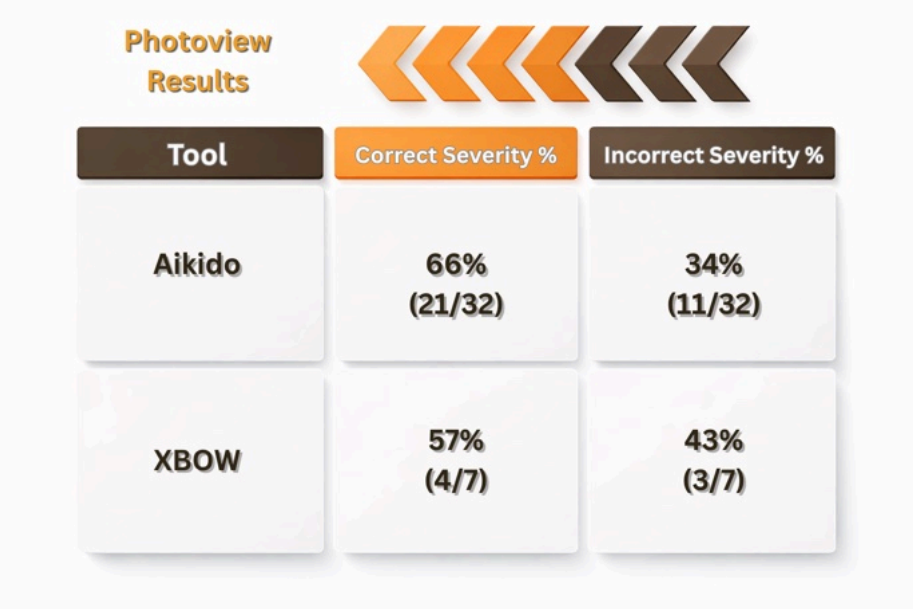
Summarized notes of interest from the review are shown below. All others should be assumed to be acceptable and not noteworthy:

- XBOW-PHOTOVIEW-2: The evidence cited in the report may suggest a file enumeration or path traversal vulnerability, but this is not the case.
- XBOW-PHOTOVIEW-3: This vulnerability is notable because it chains off the SQLi identified in a separate finding.
- XBOW-PHOTOVIEW-5: The payload present in the report PDF is truncated.

Per the XBOW platform's internal severity scoring system, these ranged from *informational* to *critical* severity. Based on our review, our team would have scored them between *informational* and *critical* as well. However, we felt one *medium* severity should be classified as *informational* severity, another should be rated as a *low* severity, and one should have been elevated to a *high* severity vulnerability.

According to our scoring:

- 57% (4/7) of the reported vulnerabilities were assigned the correct severity
- 43% (3/7) of the reported vulnerabilities were assigned an incorrect severity



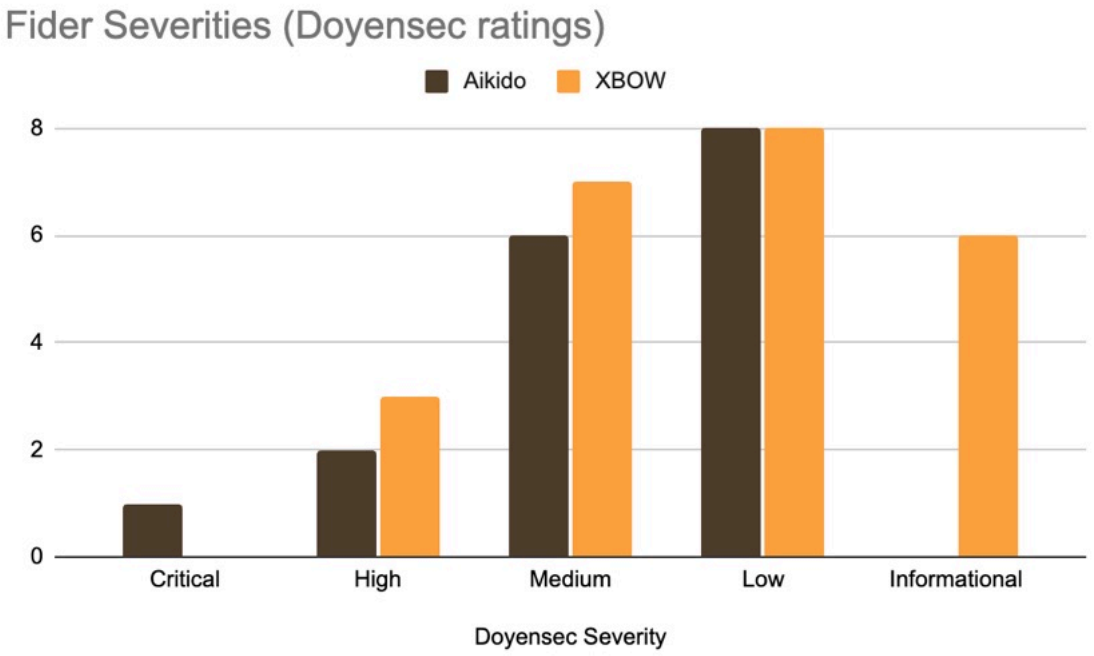
With respect to the Photoview results, the XBOW platform overstated the severity of two vulnerabilities and understated one vulnerability, per our assessment. The severities assigned were within one or two levels of our classification.

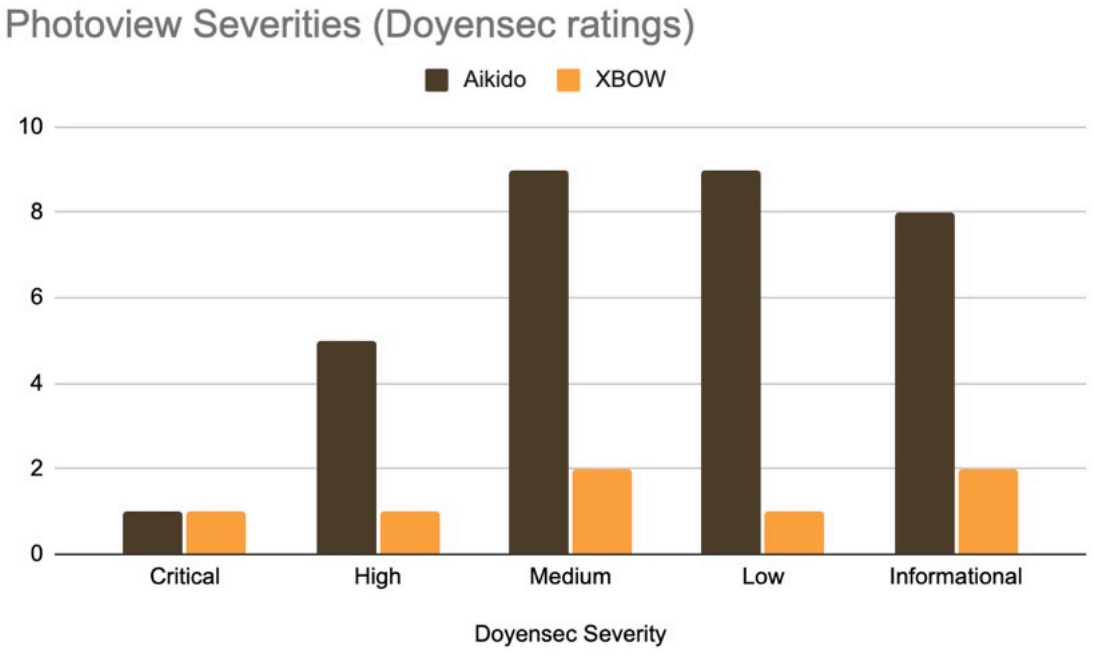
The reasoning for each is given below:

Adjusted Severities				
Finding ID	**Finding**	**Original Severity**	**Adjusted Severity**	**Reason**
XBOW-PHOTOVIEW-2	d47d9a1eeb860df8e470ea36473e43948f720cd0	Medium	Informational	The information disclosed in the error messages is of limited sensitivity, and it only reveals an absolute server path [redacted] The evidence cited in the report may suggest a file enumeration or path traversal vulnerability (..%252f..%252f..%252f..%252fetc%252fpasswd), but this is not the case.
XBOW-PHOTOVIEW-5	8d84eb78aa9d37cd219bf6d2839626b6044eb295	Medium	Low	The information disclosed in the error messages is of limited sensitivity;
XBOW-PHOTOVIEW-6	90ac2dcc5b5fde35f26667f7a4d9e6a95c7e62bb	Medium	High	The severity has been raised since the leaked information grants any user [redacted] they do not own

As previously stated, assigning severities can be somewhat subjective, so some variance is not surprising. We take no stance on whether overstating or understating severity is preferred, since both have consequences.

The charts below show the distribution of severities for each platform, according to the Doyensec severity ratings. Overall, Aikido provided a higher number of vulnerabilities at every severity level, except *Informational* (tied). Aikido also provided a slightly higher percentage of *Critical*, *High,* and *Medium*-severity vulnerabilities (1-2% each) and a 6% greater percentage of *Low*-severity vulnerabilities.


Fider Severities (Doyensec ratings)
Aikido
XBOW
8
6
4
2
0
Critical
High
Medium
Low
Informational
Doyensec Severity


Photoview Severities (Doyensec ratings)
Aikido
XBOW
10
8
6
4
2
0
Critical
High
Medium
Low
Informational
Doyensec Severity

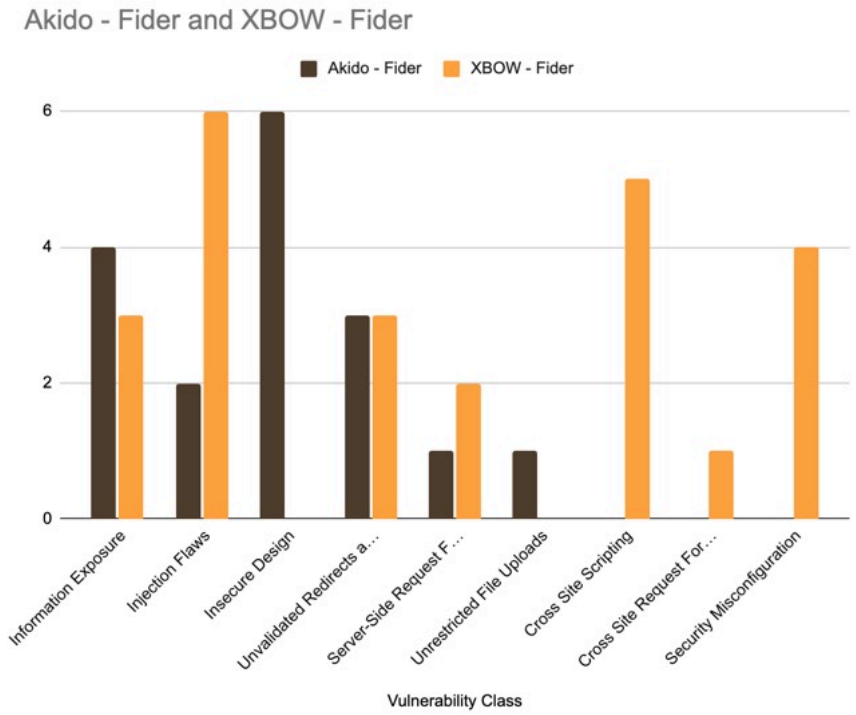
COMPARATIVE ANALYSIS

Detection Coverage

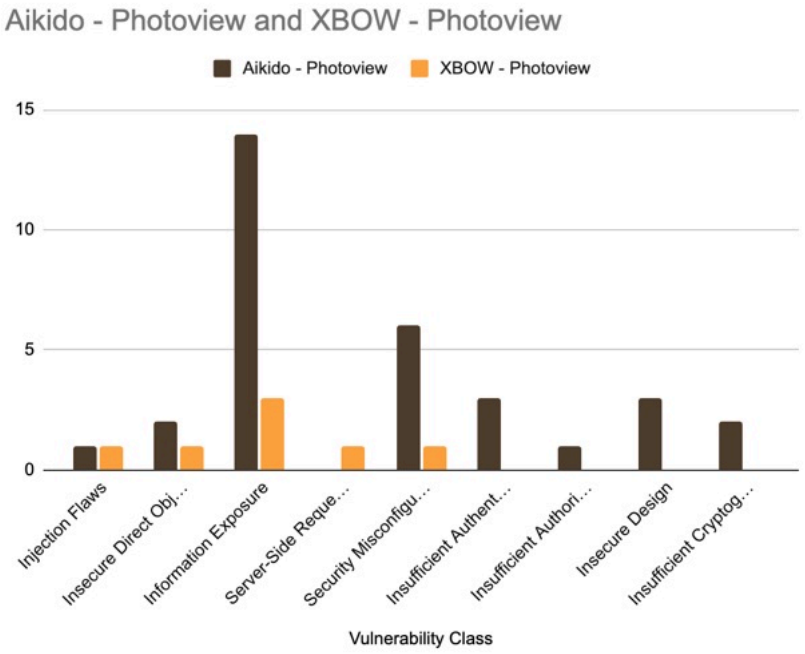
It is important to look at the distribution of vulnerability classes a platform is capable of detecting to gain insight into its breadth. Each framework *should* provide users with an overview of the vulnerability classes it can detect. This allows users to, at a minimum, be aware of the blind spots. At Doyensec, we typically assign one of the following vulnerability classes to each finding.

Vulnerability Class
Components With Known Vulnerabilities
Covert Channel (Timing Attacks, etc.)
Cross Site Request Forgery (CSRF)
Cross Site Scripting (XSS)
Denial of Service (DoS)
Information Exposure
Injection Flaws (SQL, XML, Command, Path, etc)
Prompt Injection
Insecure Design
Insecure Direct Object References (IDOR)
Insufficient Authentication and Session Management
Insufficient Authorization
Insufficient Cryptography
Memory Corruption (Buffer and Integer Overflows, Format String, etc)
Race Condition
Security Misconfiguration
Server-Side Request Forgery (SSRF)
Unrestricted File Uploads
Unvalidated Redirects and Forwards
User Privacy
Time-of-Check to Time-of-Use (TOCTOU)

The charts below show how we classified each vulnerability for each platform. This is provided to illustrate the differences in detecting each type of vulnerability that each platform demonstrated.



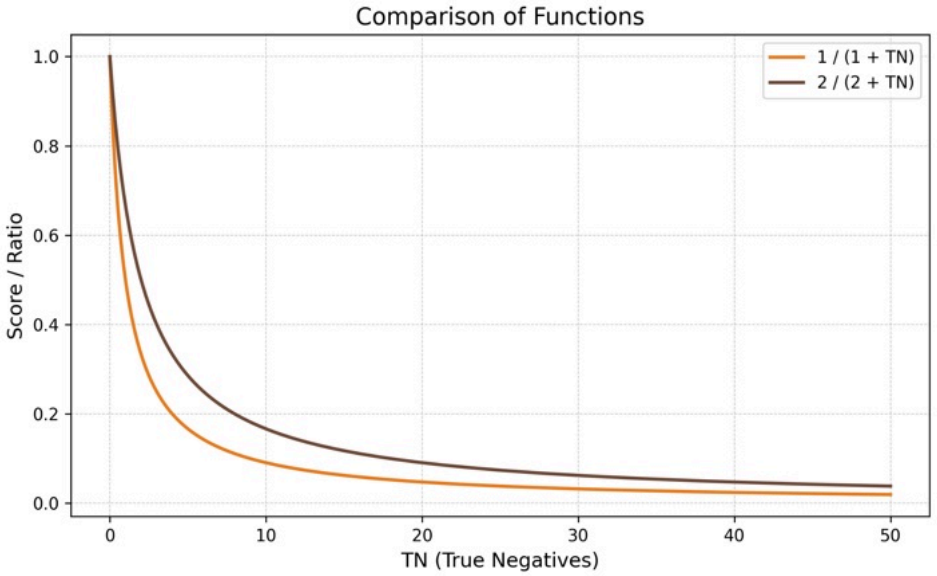
The Fider-specific chart above suggests Aikido didn't have as broad a range of detected vulnerability classes and may be more prone to finding issues typically considered as "Insecure Design". XBOW seems to have shown a broad range and possibly a greater ability in detecting common vulnerabilities like Injection Flaws and Cross-site scripting (XSS).



In the Photoview-specific chart shown above, we see the Aikido platform demonstrating the greater breadth of classes detected, with an emphasis on Information Exposure. This time, the XBOW platform appears to have demonstrated little breadth with no clear affinity for any particular vulnerability class. After looking at these distributions, the next question was how much they overlapped with each other.

Ratios

Since we did not look at the true negatives (TN), we cannot provide a true positive rate ($FPR = FP/(FP + TN)$). Further, the chart of the Aikido and XBOW functions shows that both functions exhibit identical decay behavior with increasing TN, effectively converging if you assume $TN >= 40$. In other words, the FPR for each is basically the same.



Ultimately, each organization has to decide for itself what the acceptable balance is between potentially missing vulnerabilities and wasting time chasing down non-existent vulnerabilities. So, another way to look at this without the TN could be to say what is the ratio of false positives (FP/TP), in other words, how many invalid vulnerabilities are reported as compared to true positives, where the lower value is better. For Aikido, we had $2/49 = 4\%$ versus XBOW, which had $1/31 = 3\%$.

There are, however, always caveats to raw analysis of data. As anyone who has ever run an automated security scanner can tell you, one root cause can manifest itself in numerous locations. This can cause a platform to get credit for finding multiple valid vulnerabilities, but all with the same root cause. So, teams have to ask themselves what the value is of the Nth iteration of a finding with the same root cause versus a new finding of the same type/severity with a different root cause.

For example, in the XBOW - Fider scan, we found that there were a total of 7 vulnerabilities that shared a root cause. If just those were de-duplicated (i.e., counted as 1), you would get XBOW with a ratio of $1/25 = 4\%$, which was the same as Aikido. You would, of course, also need to do

the same with Aikido's results to get the final numbers to compare. While this level of analysis wasn't part of the research scope, it's important to take all these things into consideration.

Unique Findings

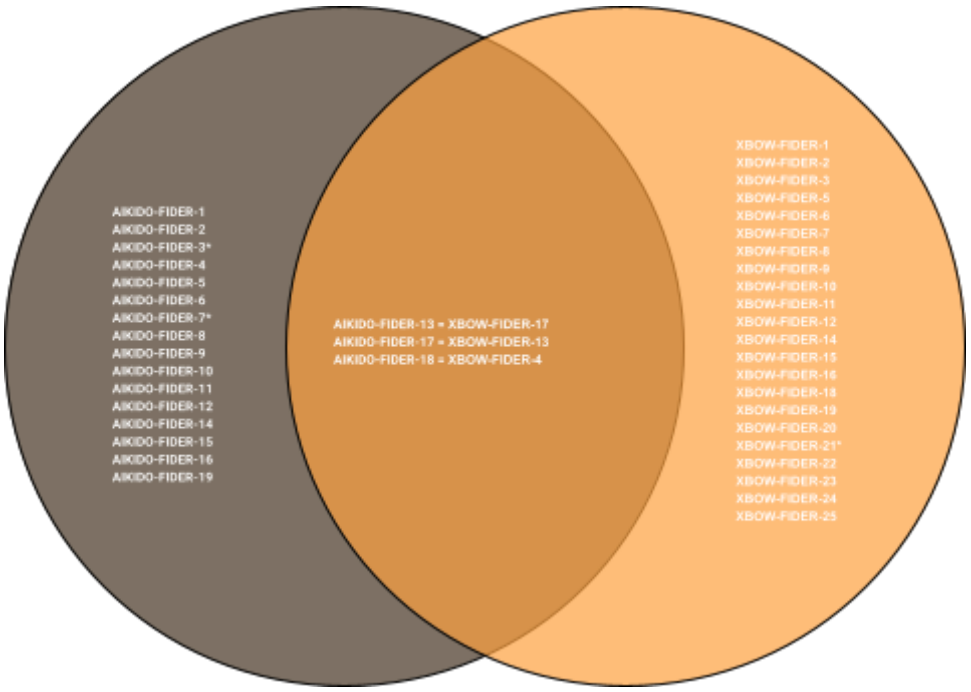
While out of scope, another angle we were interested in exploring was the overlap in findings between the platforms. We saved this until the end of our research, due to time constraints, but we were able to give it a quick review.

The diagram below illustrates the overlap between the results from both platforms for the Fider tests. As is evident, only three of the findings appear to be identical.

XBOW-FIDER-4 (M➔L) = AIKIDO-FIDER-18 (L)
XBOW-FIDER-13 (M➔L) = AIKIDO-FIDER-17 (L)
XBOW-FIDER-17 (M➔L) = AIKIDO-FIDER-13 (M➔L)

For what it's worth, we had downgraded the severity for all three on the XBOW side, from *medium* to *low* severity, as well as one on the Aikido side.

Those marked with an asterisk (*) were deemed false positives and weren't evaluated for the possibility of being duplicates.



The next diagram illustrates the overlap in results from the PhotoView tests. Here, only four of the results were deemed to be identical.

XBOW-PHOTOVIEW-1 (C) = AIKIDO-PHOTOVIEW-1 (C)
XBOW-PHOTOVIEW-2 (M➔I) = AIKIDO-PHOTOVIEW-29 (L➔I)
XBOW-PHOTOVIEW-5 (M➔L) = AIKIDO-PHOTOVIEW-27 (L)

XBOW-PHOTOVIEW-6 (M→H) = AIKIDO-PHOTOVIEW-3 (H)

Because of this low rate of **obvious** duplication, we suspected there were more that would just require digging into them more deeply to match root causes. For example, both platforms reported vulnerabilities due to missing and/or misconfigured headers. We didn't count those as the same because the headers each reported only partially overlapped.

User Experience

Qualitatively speaking, the testing with Aikido was overall a much smoother experience. The configuration was easy, with the sole exception of the previously mentioned issue, where we had to bypass security challenges for the Google user enforced by Google itself. Overall, it took less than 20 minutes to set up and start both scans. If there was any human in the loop, it wasn't noticeable to us.

On the XBOW side, the configuration was problematic and took several days, especially for the Fider testing. As detailed above in the platforms setup section, the process to kick off the scans felt very manual, since we needed to wait for a sales representative to begin, the scan itself paused/crashed multiple times, and it required over 22 emails to be exchanged with their customer support. For the Fider application specifically, the process took more than a week to complete. Also, the final deliverable was not provided immediately after testing, and instead, we had to wait five days.

LIMITATIONS OF THE STUDY

Important caveats

As with most research, it is assumed that with more data, the results become more reliable. Since our examination only included two applications, within the selection criteria given previously, it can be reasonably argued that the results may differ given a larger corpus of applications being tested.

The same could be said for the fact that we only explored one configuration of the platforms. Potentially, running more scans with different scan options would provide different results.

It is also possible that preparing the applications in different ways may have resulted in different outcomes. For example, adding more or different sample data/users, removing unused code, or testing different versions of the software could all impact the results.

We would recommend that readers treat our study as a test drive, using it as a practical reference to evaluate how these platforms perform in real-world scenarios and to determine their suitability for their own environments.

CONCLUSION

Key Takeaways

Both the Aikido and XBOW platforms proved to be capable AI-based application security testing solutions, with differences that were generally incremental rather than stark. They both demonstrated the ability to identify previously unknown vulnerabilities in low-to-medium complexity web applications developed with modern technologies. Aikido showed an advantage in the setup process, overall testing and reporting speed, and in how its testing approach affected the target application and surrounding environment. It also identified a higher number of true positives (49 versus 31) and delivered somewhat stronger reporting quality, particularly in terms of organization and clarity of replication steps. Both platforms performed comparably when it came to accurately assigning vulnerability severity. The XBOW platform, however, demonstrated a slight edge in maintaining a lower false positive count (1 versus 2) and may have enabled somewhat faster validation of findings based on its reports despite formatting issues. Overall, **the results suggest that while the Aikido platform offers a better user experience and incremental gains in efficiency and depth, the XBOW platform provides marginal benefits in precision.**

Overall Comparison		
	Aikido	**XBOW**
Setup process (e.g., ease, supported tech)	🤖	
Speed of testing and reporting	🤖	
Testing process (e.g., did things break, testing cleanup)	🤖	
More true positives (49 vs. 31)	🤖	
Less false positives (2 vs. 1)		🤖
Ratio of false/true positives (4% vs. 3%)		🤖
Correct severity assigned (69% vs. 68%)	🤖	
Higher-severity findings	🤖	
Reporting (e.g., organization, replication steps)	🤖	

FUTURE RESEARCH

We would propose a few possible future research topics. First off, expanding the research to include a larger number of applications would provide a more decisive and reliable result. Due to the costs involved in performing such a comparison, we would expect vendors to volunteer in such community-driven comparisons.

Secondly, performing a head-to-head comparison between the AI platforms and human testers at various skill levels could be used to gauge each platform's level of sophistication.

Finally, a study comparing the results of the AI platform to various SAST/DAST tools could clarify what, if any, improvement these platforms represent over the previous generation of options.

APPENDIX A

The following resources are provided for readers interested in replicating our results or evaluating our approach. Due to the SaaS nature of both AI platforms, Doyensec did not have internal visibility into the underlying models or their specific versions at the time of testing.

File Manifest		
File name	**Description**	**Hash (SHA-256)**
oss_webapps_dataset.csv	A CSV containing a list of 442 web applications used to select targets from	61a395a8bfabf22e25f50713b0a107ce4a4a0a05b27c7fa7aac3490e8a2f4c9e
XBOW_Aikido_Findings_Results_PUBLIC.xlsx	All of the findings from both platforms' scans of both apps	a639ed23ed72d4d04bb4ee58f31c45b15901ce3ddeefdc7450614e1b581d41b6
fider-docker-compose.yml	Docker compose file used for the Fider deployment	5dd4aa0f108e25f3bc1543127e4da3e929966a49555c50257a5fda4c485f5d46
photoview-nginx.conf	Nginx configuration for the Photoview deployment	a85653572bfed4e76f8809cbf84f5ca709976e7c43149c4badf08a3500f8c2d0
photoview-docker-compose.env	Environment variables for Docker - used with Photoview deployment	9000d4e20b010772263dbf18b9c412006e10b8d4ac42470ea26363f812d05b13

ABOUT DOYENSEC

Doyensec was founded in 2017 by John and Luca who are its only stakeholders. The company exists to further the passion and focus of its creators. We aim to provide research-driven application security, enabling trust in our client's products and evolving the resilience of the digital ecosystem.

With offices in the US and Europe, Doyensec has access to a unique talent pool of security experts capable of providing worldwide consulting services.
We keep a small dedicated client base and expect to develop long term working relationships with the projects and people involved. We will find bugs, but we know that is just the first step in the process. At any stage of your security maturity, you can rely on Doyensec to solve your unique application security needs.

We value and rely on the following principles:

- **Passion**. We believe quality comes from passion and care. We love what we do, and continuously work on mastering our craft. Every engagement is finely executed with dedication and attention to details.
- **Expertise**. Our team has decades of experience in application security. We are industry leaders in penetration testing, reverse engineering, and source code review. Doyensec researchers have discovered numerous vulnerabilities in widely-deployed products, secured fortune 500 enterprises, advised startups and worked with tech companies to eradicate security flaws.
- **Focus**. Security craftsmanship is all about individual attention and delivering tailored security services and products. We concentrate on application security and do fewer things, better.
- **Research**. The fast changing landscape of technologies and security threats requires constant innovation. We are dedicated to providing research-driven application security and therefore invest 25% of our time in building security testing tools, discovering new attack techniques and developing countermeasures.

ABOUT THIS RESEARCH WORK

This research is based upon work financially supported by Aikido. Despite that, Doyensec had complete freedom on the research execution and publication. While Aikido had the right to decide on whether to publish the effort in its entirety or just citing parts according to our Citation Guideline, under no circumstances has Doyensec adjusted the results represented in this publication.


DOYENSEC