Introduction

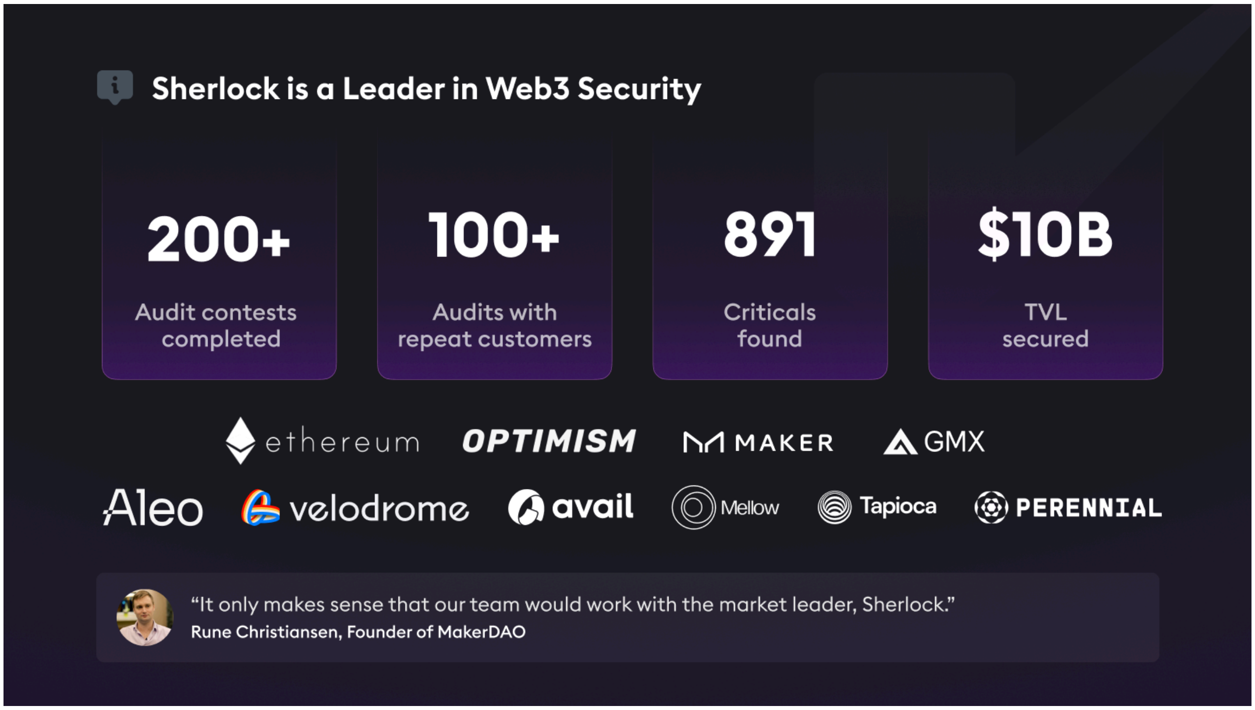
Sherlock is a leading Web3 security company, providing a full suite of services designed to safeguard protocols at every stage of development. Our offerings include audit contests, collaborative audits, bug bounty management, and exploit coverage through Sherlock Shield. This comprehensive approach enables us to deliver tailored solutions that address the unique security challenges of each project.

Our extensive experience and proven success are highlighted by:

- Over 230 audits were completed, with a 98% success rate in finding at least a medium-severity vulnerability.
- Over $1.5mm was awarded to Sherlock to manage multiple security grant programs by some of the largest protocols.
- We executed one of the *largest audit contests in history* for MakerDAO with a value of $1.35mm.
- Multiple case studies highlighting the effectiveness and efficiency of our services:
    - Tokemak Case Study
    - Index Coop Case Study
    - Ajna Finance Case Study
    - Perennial Case Study

Sherlock's consistent track record of exceeding expectations has fueled high demand for our audits, supported by the number of projects that return for a **second** audit after their initial success. This list includes:

Ethereum Foundation, GMX, MakerDAO, Velodrome, Mellow, Optimism, Ajna, LooksRare, Gitcoin, Index Coop, Rio Network, Opyn, Notional, OlympusDAO, Lyra, Perennial, Sentiment, Symmetrical, BOND Protocol, Merit Circle, DODO, JOJO, NounsDAO, Footium, Illuminate, Union Finance, Unstoppable/Alchemix, WAGMI, Blueberry, Telcoin, Cooler, Teller, Sense and many others.



Sherlock is a Leader in Web3 Security

200+
Audit contests completed

100+
Audits with repeat customers

891
Criticals found

$10B
TVL secured

ethereum
OPTIMISM
MAKER
GMX
Aleo
velodrome
avail
Mellow
Tapioca
PERENNIAL

"It only makes sense that our team would work with the market leader, Sherlock."
Rune Christiansen, Founder of MakerDAO

Why Sherlock for GMX-Solana?

Building on Recent GMX-Solana Audit Success

Having already been selected to audit GMX-Solana's codebase once, Sherlock is uniquely positioned to conduct a second audit with unmatched efficiency. We are handpicking the top-performing auditors from the first audit (which we completed 12/4/24)), ensuring that we leverage their deep familiarity with your codebase. By assembling a team of experts who have already demonstrated their ability to identify vulnerabilities in GMX-Solana's system, our collaborative audit will deliver a focused, thorough, and highly effective review.

During the first audit, Sherlock:

- Conducted a thorough review over the course of a month, gaining a deep understanding of your architecture and specific security requirements.
- **Identified 6 High and 11 Medium vulnerabilities**, ensuring GMX-Solana's platform remained robust and resilient.
- Utilized a team-based audit approach, assembling a group of security experts with unparalleled experience to collaborate in reviewing the codebase and identifying vulnerabilities.

For this second audit, we've selected the two top-performing researchers from the first engagement, ensuring continuity and leveraging their proven expertise with your codebase. This strategic choice will allow us to deliver precise, actionable insights with maximum efficiency.

A Historical Track Record of Delivering Results for GMX

Sherlock has a long-standing relationship with GMX, having successfully completed two prior audits on different scopes before our most recent GMX-Solana engagement. In February 2023, during our first audit for GMX, we identified 21 High-severity vulnerabilities and 38 Medium-severity vulnerabilities, delivering critical insights that significantly strengthened the security of the protocol.

Building on this success, we conducted an update audit in May 2023, where we found an additional 5 High-severity vulnerabilities and 11 Medium-severity vulnerabilities. These results highlight our ability to uncover key issues across varying scopes while ensuring the highest standards of security.

Now, having just completed another audit of GMX-Solana, Sherlock is positioned well to leverage our deep familiarity with GMX's systems and processes. Our extensive history with the protocol, combined with the continuity of expertise we bring through our collaborative audits, ensures that we can deliver another round of exceptional results tailored to your needs.

Expert Security Researchers Selected

Lead Security Researchers

Bin2chen

Availability: Starting January 1st

Experience:

- **Rust Expertise:** bin2chen is proficient in Rust and has scored multiple top positions in Rust-based audits, demonstrating his ability to effectively identify security vulnerabilities in Rust codebases. He applied this expertise in the initial GMX-Solana audit engagement, where he identified high and medium severity vulnerabilities.
- **Audit Experience and Success:** He has participated in numerous audit contests, achieving impressive results, including scoring first-place four times, and identifying 11 bugs in the Andromeda contest. Additionally, he has found high-severity issues in contests such as EigenLayer and has a strong track record in competitions like zkSync, Astaria, Sentiment V1, and Illuminate.
- **Derivative Protocol Expertise:** In addition to his diverse protocol knowledge, bin2chen has also audited GMXv2 on Sherlock as well as other derivative protocols, including, Perennial, and Symmetrical, showcasing his ability to identify security vulnerabilities in complex financial instruments and mechanisms.
- **Why are they a good fit:** bin2chen's expertise in Rust, combined with his impressive audit experience & diverse protocol knowledge make him a strong candidate to audit the GMX Solana codebase. His prior experience with the GMX Solana codebase also gives

him insight into areas that require more attention, such as module interactions and dependencies. His skills and experience, including his expertise especially in auditing derivative protocols, will enable him to effectively identify and report security vulnerabilities in the protocol.

Bin2Chen’s thoughts after the first GMX-Solana audit:

Summary: *The project is well structured and has been designed to avoid code duplication by using common abstractions in `` `create/close/validate/*TransferInOperation/*TransferOutOperation/*MarketOperation` `` and so on. The overall code base is robust and extensible. However, abstract structure is particularly important for setting parameters of each call and subtle differences in different scenarios. Some issues were found in these details.*

Follow-up Areas to Focus: *The project has a large amount of code, high complexity, and limited time. Currently, the audit is being conducted by module division. During the audit, calls to other modules (e.g., exchange/pnl/pool value calculations) are involved, and due to time constraints, an in-depth analysis is not feasible, which may pose some risks.*

Recommendations for Next Steps: *More time per auditor is needed to analyze module interactions and dependencies. Assigning at least two auditors per module is recommended to improve coverage.*

IIIIIII

Availability: Starting January 1st

Experience:

- **GMX Expertise:** lllllll has extensive experience with GMX, having led the contest for GMXv2 and the GMX Update contest, scoring first place on both and identifying 16 critical vulnerabilities. He also participated in the initial GMX Solana audit engagement, where he found security issues due to small oversights in the porting of the Solidity code to Rust/Solana. This demonstrates his in-depth knowledge of the protocol and its potential security risks.
- **DeFi Auditing Experience:** He has also audited other decentralized finance (DeFi) protocols, including perpetual exchanges (e.g., Perpetual), Liquid Staking Protocols (e.g., FRAX, StakeHouse), and vesting protocols (e.g., Rio Vesting Network), showcasing his broad expertise in the field.
- **Why are they a good fit:** lllllll's exceptional track record with GMX, combined with his experience auditing similar DeFi protocols and his broad skill set, make him an ideal candidate to audit the GMX Solana protocol, ensuring the security and integrity of the new implementation. His prior experience with the GMX Solana codebase also enables him to identify potential integration issues and areas that require more attention.

IIIIIII’s thoughts after the first GMX-Solana audit:

Summary: *The two areas I covered during the review were gmsol_utils and gmsol_model. Due to the significant number of lines we had to cover in the short amount of time we had, and the complexity of the code itself, I spent approximately 12+ hours a day (compared to the usual eight) reviewing and understanding the code, for the duration of the review. I found the code to be complex, but well-organized and well-written. The code was fun to read, and I enjoyed comparing the design decisions and tradeoffs between the Rust and Solidity versions. Most of the security issues I found were due to small oversights in the porting of the Solidity code to Rust/Solana, rather than being due to structural/organizational or logical errors.*

Follow-up Areas to Focus: *Given the limited amount of time to cover my area, my focus was on the implementation details and finding bugs within the model, rather than looking for integration issues in the caller of the model. When there was confusing behavior, or behavior that differed between the Solidity version and the Rust version, I would look outside of the model for potential bugs. If there was no difference/bug within the model, I didn't have enough bandwidth to examine whether the callers were integrating properly.*

One concrete example that comes to mind is the fact that for the position decrease code, there is a method that lets the caller change the swap kind from the default of NoSwap, to something else. There happened to be a porting difference where swapping is forced, regardless of the swap kind requested, so that caused me to look at callers of the setter of the swap kind to see if there was a bug there. To my surprise there were no callers of the function, which raised questions about whether there could be problems where the Solidity code allowed not swapping, whereas the current code required it. I found a couple of these issues based on knowledge I gained from previously auditing the GMX Solidity code base, but given that there are other differences/tradeoffs that did not involve my layer, it's hard to say whether there aren't other integration issues lurking in that or similar areas.

Recommendations for Next Steps: *My recommendation would be to do another, longer audit, and have the integrations between the various other layers specifically be examined. Each auditor would examine the layer above or below the one they last reviewed, in order to find missed integration issues.*

Scope Overview and Approach

In the initial review, IIIIIII covered the entirety of the `gmsol-utils` and `gmsol-model` crates. His strategy to ensure comprehensive code review without overlooking any areas followed a bottom-up approach. Since Rust avoids complex inheritance hierarchies, its program structure tends to be flatter, with dependencies being more explicit and self-contained. This makes verifying lower-level modules more manageable before evaluating their integration into higher layers. For each in-scope file, IIIIIII examined all `use` expressions, expanded and normalized them into a single expression, and then reordered the files based on dependency hierarchy—starting with files that had no dependencies on later ones. In some instances, he needed to analyze higher-level files to understand the context (e.g., determining the contents of each pool in BaseMarket, which was time-intensive). Overall, the approach was effective.

The approach lllllll would take for the new scope remains the same. lllllll will collaborate with bin2chen, who previously reviewed the GLV, GT, Utils, and Access Control sections of the codebase. Given the priority of the timelock and treasury programs, bin2chen will begin with the treasury module, as it references parts of the GT code he covered earlier. Meanwhile, lllllll will start with the timelock module, which has minimal dependencies. Once they complete their respective areas, they will review each other's work before tackling the remaining sections of the gmsol-store code within the available time.

They will work on different modules initially to avoid overlapping and competing for findings in the same sections. Both will ultimately review the same files, ensuring that one can catch anything the other might miss. With a total of 14,750 SLOC to be audited, and considering lllllll's previous pace of 350 SLOC per 12+ hours (or 233 SLOC per 8 hours), the optimal duration for a comprehensive review is approximately 82 days. This includes a daily target of 208 SLOC for 71 days and an additional 11 days for investigation and report writing.

For this review, scheduled over 7 weeks, the auditors will allocate 6 weeks to review at a rate of 208 SLOC/day, covering between 9,000 and 10,000 SLOC. The final week will be reserved for investigations and documenting the findings. The timelock and treasury modules comprise 2,916 SLOC, leaving 11,834 SLOC for the remaining gmsol-store code. To maximize coverage of code from the model crate, the auditors will adopt a bottom-up approach, focusing first on each layer interacting with the model and then proceeding to subsequent layers for the remainder of the audit.

Sherlock Collaborative Audit

Audit Structure & Process

A Sherlock collaborative audit brings together the industry's top security researchers to work together in identifying bugs in the codebase. GMX-Solana stands to benefit significantly from this collaboration due to the complexity of its codebase. Auditors familiar with the GMX-Solana codebase will have the opportunity to collaborate closely with Rust and Solana experts to discover unique and hard-to-detect bugs.

What it Includes

- The team composition of the collaborative audit will be 2 Lead auditors who performed best on the first audit of GMX-Solana.
- Bugs will be shared by the auditors as soon as they are found.
- Assistance with bug fixes.

- A three-day mitigation review of all the fixes done by the auditors.

Proposed Timeline

Timeline: Proposed 7 - week Audit

- **Phase 1**: Audit (Jan 6th - Feb 24th)
- **Phase 2:** Bugs are being fixed by GMX-Solana team (Jan 6th - Mar 3rd)
- **Phase 3**: Fix Review and Audit Report (Mar 3rd - Mar 10th)

***Proposed timing is subject to change based on the stated needs of GMX-Solana and the number of bugs found during the audit*

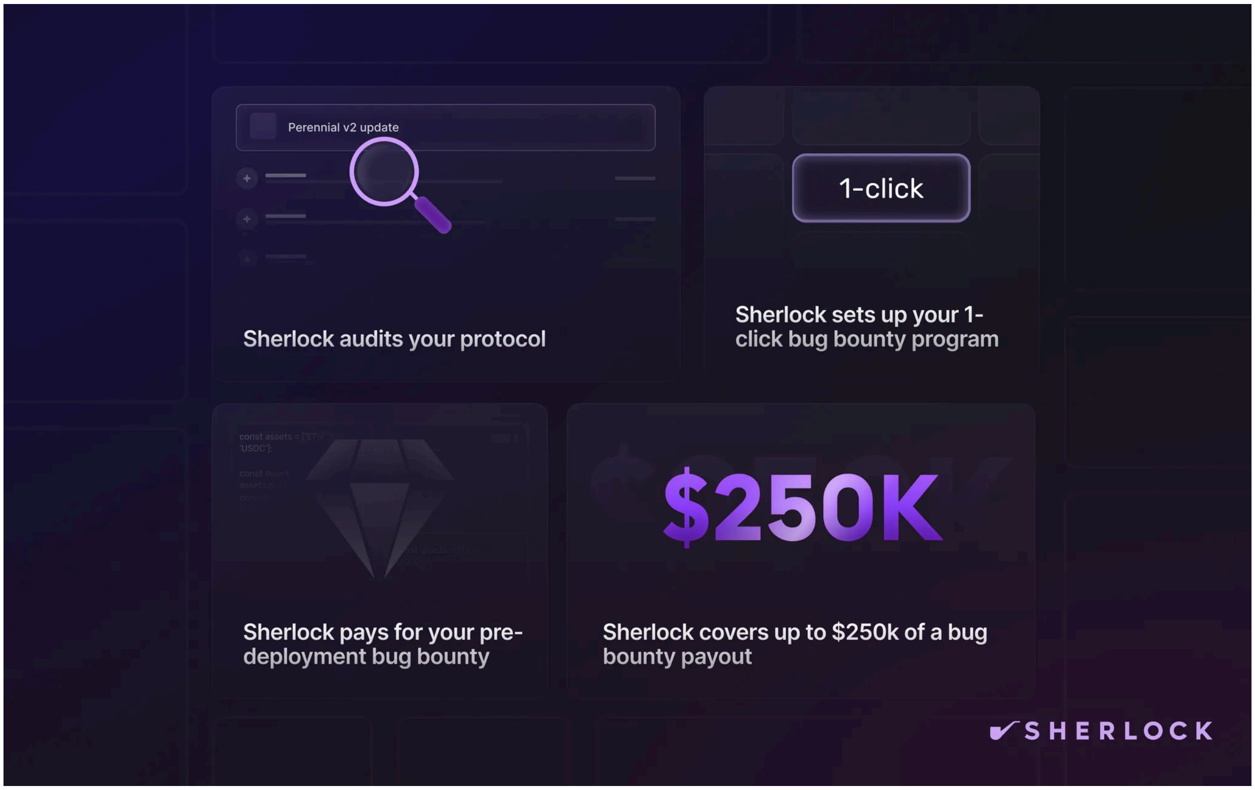
Pricing

Criteria	**Amount**
IIIIIII Pay	$175,000
Bin2chen Pay	$105,000
Sherlock Fee	$20,000
Total	$300,000

Bug Bounty & Exploit Coverage (Optional)

Earning the trust of liquidity providers and users is one of the biggest challenges when introducing a new product in the Web3 market. Additionally, Sherlock believes there is a lack of incentive for auditors in the industry to perform at the highest level.

To address these issues, GMX-Solana can establish a bug bounty and for a monthly fee, Sherlock will cover the costs in the event of a valid critical submission. This bug bounty program and its coverage can be announced and launched concurrently with the deployment of GMX-Solana to mainnet.



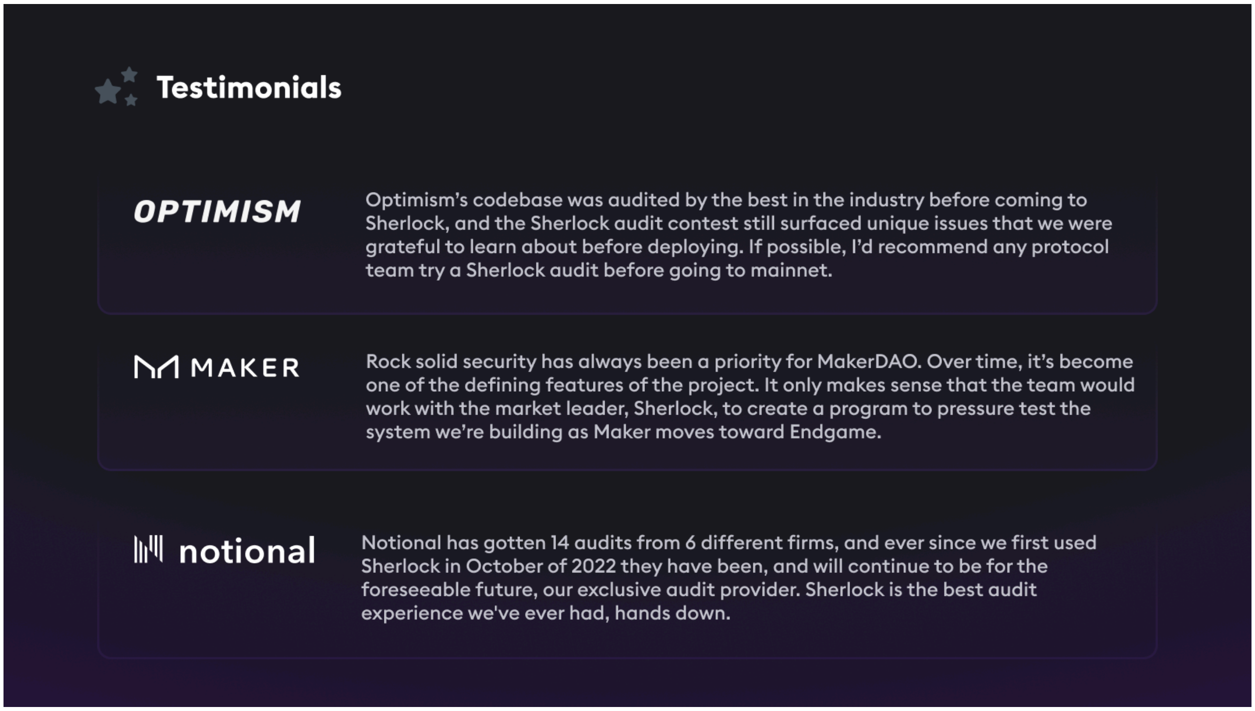
Sherlock audits your protocol

Sherlock sets up your 1-click bug bounty program

Sherlock pays for your pre-deployment bug bounty

Sherlock covers up to $250k of a bug bounty payout

Testimonials



Testimonials

OPTIMISM

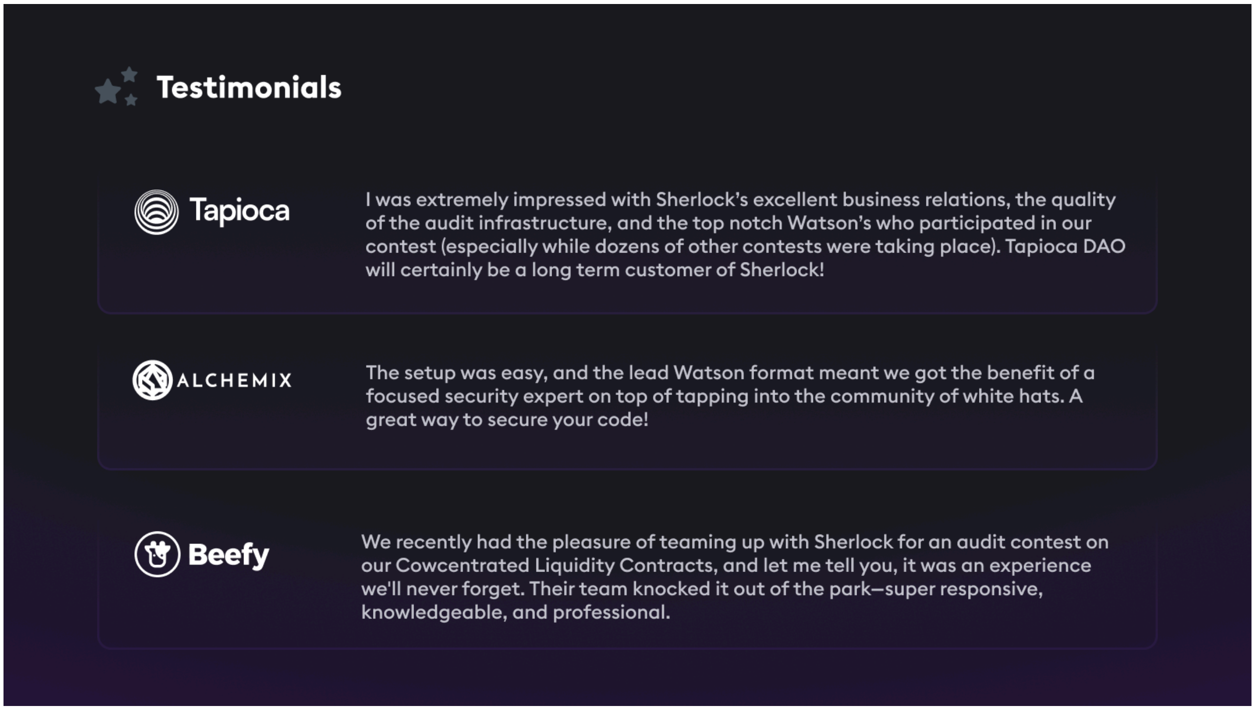
Optimism's codebase was audited by the best in the industry before coming to Sherlock, and the Sherlock audit contest still surfaced unique issues that we were grateful to learn about before deploying. If possible, I'd recommend any protocol team try a Sherlock audit before going to mainnet.

MAKER

Rock solid security has always been a priority for MakerDAO. Over time, it's become one of the defining features of the project. It only makes sense that the team would work with the market leader, Sherlock, to create a program to pressure test the system we're building as Maker moves toward Endgame.

notional

Notional has gotten 14 audits from 6 different firms, and ever since we first used Sherlock in October of 2022 they have been, and will continue to be for the foreseeable future, our exclusive audit provider. Sherlock is the best audit experience we've ever had, hands down.



Testimonials

Tapioca

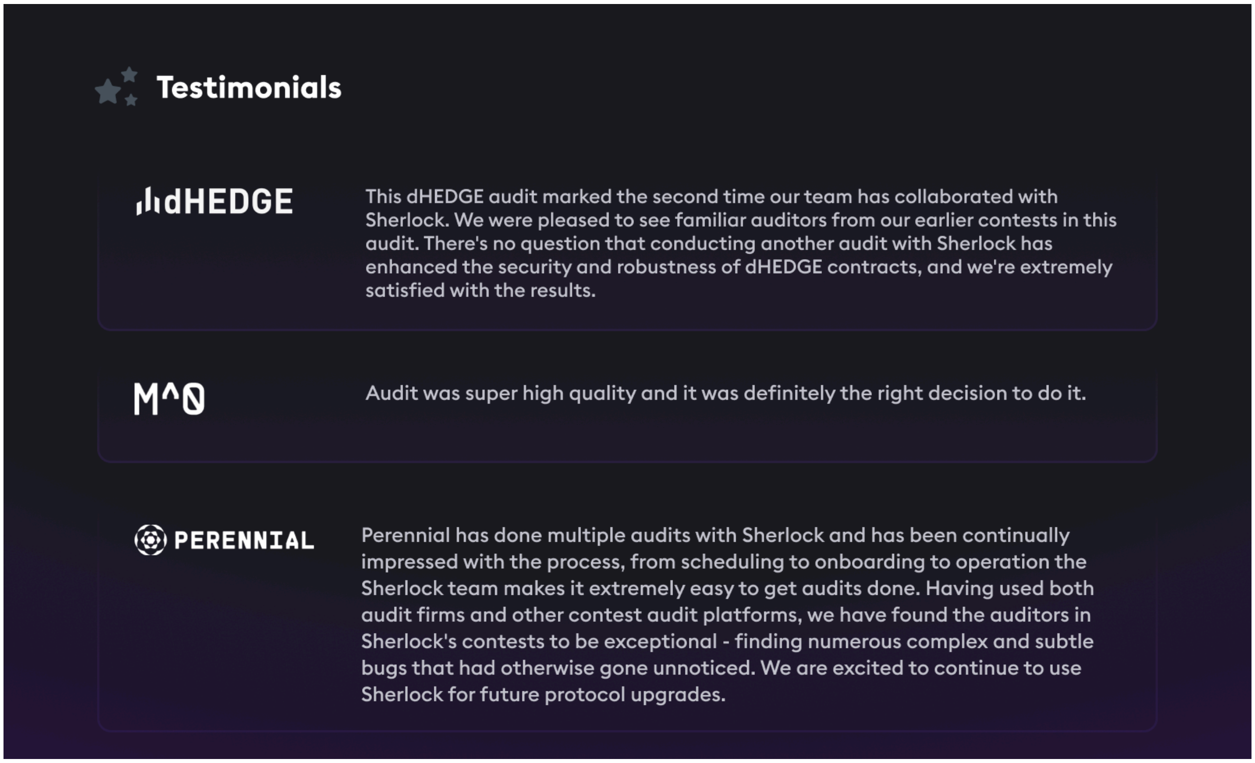
I was extremely impressed with Sherlock's excellent business relations, the quality of the audit infrastructure, and the top notch Watson's who participated in our contest (especially while dozens of other contests were taking place). Tapioca DAO will certainly be a long term customer of Sherlock!

ALCHEMIX

The setup was easy, and the lead Watson format meant we got the benefit of a focused security expert on top of tapping into the community of white hats. A great way to secure your code!

Beefy

We recently had the pleasure of teaming up with Sherlock for an audit contest on our Cowcentrated Liquidity Contracts, and let me tell you, it was an experience we'll never forget. Their team knocked it out of the park—super responsive, knowledgeable, and professional.



Testimonials

dHEDGE

This dHEDGE audit marked the second time our team has collaborated with Sherlock. We were pleased to see familiar auditors from our earlier contests in this audit. There's no question that conducting another audit with Sherlock has enhanced the security and robustness of dHEDGE contracts, and we're extremely satisfied with the results.

M^0

Audit was super high quality and it was definitely the right decision to do it.

PERENNIAL

Perennial has done multiple audits with Sherlock and has been continually impressed with the process, from scheduling to onboarding to operation the Sherlock team makes it extremely easy to get audits done. Having used both audit firms and other contest audit platforms, we have found the auditors in Sherlock's contests to be exceptional - finding numerous complex and subtle bugs that had otherwise gone unnoticed. We are excited to continue to use Sherlock for future protocol upgrades.