The Hack Library

A Compilation of Notable TestMachine Hacks

05 August 2024

Introduction

1. / Overview

In the rapidly evolving world of decentralized finance (DeFi) and blockchain technology, the security of smart contracts has become a paramount concern. As the value locked in DeFi protocols continues to grow, so too does the incentive for malicious actors to exploit vulnerabilities and steal funds. In this context, TestMachine has emerged as a leading provider of advanced smart contract security solutions, leveraging techniques in artificial intelligence and reinforcement learning to identify and mitigate vulnerabilities before they can be exploited.

This Hack Library documents the effectiveness and impact of TestMachine's approach, showcasing a selection of hacks that our platform has successfully executed in simulated environments. By independently discovering these hacks, TestMachine demonstrates its ability to identify and exploit vulnerabilities that have led to significant financial losses and disruption in the DeFi ecosystem.

One of the most notable examples featured in this library is the Euler Finance hack of 2023, which resulted in the loss of over $200 million in user funds. TestMachine's Predator platform was able to identify and exploit the key vulnerabilities that enabled this hack. By identifying these issues in a simulated environment, TestMachine showcased its potential to prevent similar incidents from occurring in the future.

In the following sections, we will delve into each of the featured hacks in more detail, providing a technical overview of the underlying vulnerabilities, the steps taken by TestMachine to recreate the exploit, and the key lessons learned from each incident.

Methodology

2. / How Does TestMachine Work?

TestMachine's Predator platform employs an AI-driven approach to smart contract security auditing, leveraging advanced techniques from the field of reinforcement learning (RL). Our methodology centers around the concept of "*campaigns*" – focused, iterative testing processes that allow our RL agents to explore and interact with a target smart contract in a simulated environment.

At the core of each campaign are two key components: target invariants and reward functions. Target invariants are specific properties or conditions that should always hold true for the contract to be considered secure and functioning properly. These invariants are carefully defined in collaboration with our clients, considering the unique requirements and constraints of their particular use case.

Reward functions, on the other hand, are mathematical expressions that guide the learning process of our RL agents. These functions assign positive rewards for actions that bring the contract closer to violating an invariant, and negative rewards for actions that maintain or reinforce the invariant. By carefully crafting these reward functions, we can incentivize our agents to find the most efficient paths to identifying and exploiting potential vulnerabilities.

During a campaign, our RL agents interact with the target contract in a simulated environment, exploring the space of possible actions and observing the resulting contract states. As they do so, they learn to optimize their strategies based on the feedback provided by the reward functions, becoming increasingly efficient at identifying and exploiting potential vulnerabilities.

Throughout the campaign, we monitor and analyze various metrics and indicators, such as the distribution of function calls, the frequency of reverts, and the exploration of unique contract states. These insights allow us to assess the progress and effectiveness of our testing process, identify areas of concern, and adapt our approach as needed.

At the conclusion of each campaign, we generate a comprehensive report detailing our findings, including any identified vulnerabilities, their potential impact, and recommendations for remediation. We also provide a range of visual aids and analytical tools, such as function call histograms and state transition graphs, to help our clients better understand the results of our testing process.

By combining the power of AI and reinforcement learning with a rigorous, systematic approach to defining invariants and reward functions, TestMachine's Predator platform offers unparalleled insights into the security and reliability of smart contracts. Our methodology enables us to identify and mitigate vulnerabilities that would be missed by traditional auditing techniques, providing our clients with the highest level of assurance and protection for their digital assets.

3. / Euler Finance

3.1. Context

Euler Finance, a prominent decentralized finance (DeFi) lending protocol, suffered a significant exploit in March 2023. The hack resulted in the loss of approximately $196 million in cryptocurrency assets. The vulnerability stemmed from a subtle logic error in the protocol's smart contract that allowed a malicious actor to accumulate bad debt and subsequently liquidate their own position, effectively draining tokens from the protocol's vault.

Table 1: Details of Euler Finance exploit.

Vulnerability	Business Logic Error
Time to Exploit	40 hours
Network	Ethereum Mainnet
Novelty of Attack Vector	Rediscovered by Predator

3.2. How Predator Did It

Predator, TestMachine's AI-driven smart contract auditing system, successfully replicated the Euler hack, demonstrating its ability to identify complex vulnerabilities that eluded human auditors. The system's approach highlighted several key advantages over traditional auditing methods:

1. Multi-agent simulation: Predator utilized two simulated Externally Owned Accounts (EOAs) in its campaign. This multi-agent approach allowed it to circumvent Euler's security measures, which were designed to prevent a single account from both taking out a loan and liquidating its own position. By using one EOA to borrow and another to liquidate, Predator exploited a blind spot in the protocol's security logic.
2. Identification of complex interaction patterns: The vulnerability in Euler's protocol arose from the intricate interplay between different contract functions. Predator's reinforcement learning algorithms excel at exploring vast state spaces and identifying non-obvious sequences of actions that can lead to exploitable states. This capability allowed it to discover the specific sequence of transactions that could breach the protocol's security.
3. Simplified exploit execution: While the original Euler hack employed a flash loan, requiring the attacker to bundle a complex sequence of transactions as raw `calldata`, Predator demonstrated a more straightforward approach. The system's simulated agents were initialized with sufficient funds, eliminating the need for borrowing. This simplification allowed Predator to

execute the exploit through a series of individual transactions, which could later be adapted into a flash loan attack if needed.

4. Rapid iteration and learning: Unlike human auditors who might be constrained by time or cognitive limitations, Predator can rapidly iterate through thousands of potential attack vectors. This exhaustive exploration allows it to uncover vulnerabilities that might be overlooked in manual code reviews or more targeted testing approaches.
5. Invariant violation detection: Predator's ability to identify and exploit this vulnerability stemmed from its focus on violating key protocol invariants. In this case, the system recognized that the separation between borrower and liquidator accounts was a critical invariant that, if breached, could lead to unauthorized token extraction from the protocol's vault.

By replicating the Euler hack, Predator demonstrated its capacity to identify sophisticated vulnerabilities in complex DeFi protocols. This case underscores the potential of AI-driven security auditing tools to complement and enhance traditional smart contract auditing processes, potentially preventing significant financial losses in the future.

4. / Deus Dao

4.1. Context

Deus Dao, a decentralized finance (DeFi) protocol, experienced a significant security breach due to a critical vulnerability in their DEI token contract. The core issue lay in an improperly configured `burnFrom` function, which is typically used to allow approved addresses to burn tokens on behalf of token holders. However, due to a programming oversight, this function inadvertently allowed any external actor to manipulate the allowance parameter of any token holder, creating a severe security risk.

Table 2: Details of Deus Dao exploit.

Vulnerability	Vulnerable Function Logic
Time to Exploit	1 Minute
Network	Arbitrum Mainnet
Novelty of Attack Vector	Rediscovered by Predator

4.2. How Predator Did It

TestMachine's AI-driven smart contract auditing system, Predator, successfully identified and exploited the vulnerability in the Deus Dao protocol, showcasing its advanced capabilities in detecting and leveraging complex smart contract flaws:

1. Comprehensive contract analysis: Predator agents conducted a thorough examination of the DEI token contract, scrutinizing each function for potential vulnerabilities. This systematic approach allowed it to identify the misconfiguration in the `burnFrom` function, which might have been overlooked in less exhaustive auditing processes.
2. Rapid vulnerability exploitation: Upon discovering the flaw in the `burnFrom` function, Predator swiftly devised and executed an exploit strategy. The system's ability to quickly transition from vulnerability identification to exploitation demonstrates its effectiveness in simulating real-world attack scenarios.
3. Token theft simulation: In the simulated environment, Predator leveraged the vulnerability to manipulate allowances and subsequently steal tokens from existing holders. This action mimicked the real-world impact of the vulnerability, providing valuable insight into the possible consequences of such an exploit.
4. Invariant violation detection: Predator's exploit highlighted a critical violation of token contract invariants. The ability for any address to modify allowances without proper authorization

represents a fundamental breach of expected token behavior. Predator's focus on identifying such invariant violations enables it to uncover subtle yet critical flaws in smart contract logic.

5. Complex interaction pattern recognition: The vulnerability in the Deus Dao protocol involved understanding the intricate relationships between token allowances, burning mechanisms, and access controls. Predator's advanced algorithms excel at learning how to recognize these complex interaction patterns, allowing it to identify vulnerabilities that may not be immediately apparent through manual code review or traditional testing methods.
6. Efficient state space exploration: Predator's reinforcement learning algorithms enable it to efficiently explore a vast number of potential contract states and interaction sequences. This capability allows it to uncover vulnerabilities that might only manifest under specific, hard-to-predict circumstances.
7. Automated exploit generation: Once the vulnerability was identified, Predator autonomously generated the necessary transactions to exploit the flaw. This automated approach to exploit creation demonstrates the system's potential to not only identify vulnerabilities but also to assess their practical exploitability and potential impact.

Predator demonstrated its ability to identify and exploit subtle yet critical vulnerabilities in DeFi protocols. This case underscores the importance of rigorous, AI-assisted smart contract auditing, particularly for protocols with functions that involve sensitive operations like allowance modifications and token burning. The speed and effectiveness with which Predator identified and exploited this vulnerability highlight the potential of AI-driven tools to enhance blockchain security and prevent significant financial losses in the DeFi ecosystem.

5. / Socket

5.1. Context

Socket, a cross-chain interoperability protocol, suffered a significant security breach due to a vulnerability in a newly introduced router contract. This contract, which integrated with Socket's gateway contract, contained a critical flaw in its input validation process. Specifically, an unchecked data field could be forwarded through the gateway using a delegatecall, allowing for the execution of arbitrary transactions with the privileges of the router contract. This vulnerability enabled attackers to execute transferFrom transactions as if they were initiated by the privileged router contract, resulting in the unauthorized draining of tokens from users who had granted infinite approvals to the system.

The exploit's impact was substantial, with up to $656,000 in USDC drained from a single wallet. Data analytics firm PeckShield attributed the vulnerability to incomplete validation of user input, noting that users who had approved the vulnerable SocketGateway contract were particularly at risk. The malicious gateway was added merely three days before the exploit occurred, highlighting the rapid exploitation of newly introduced vulnerabilities.

The incident was further complicated by subsequent phishing attempts. Scammers, exploiting the confusion following the initial hack, used a fake Socket account to distribute links to malicious applications, urging users to revoke their approvals through another compromised app.

This hack emphasizes the critical security challenges faced by cross-chain bridges and interoperability protocols. While these systems play a vital role in facilitating interactions between different decentralized protocols, they have increasingly become prime targets for malicious actors. The Socket incident adds to a growing list of significant decentralized finance exploits targeting cross-chain bridges in recent years.

Table 3: Details of Socket exploit.

Vulnerability	Data Injection
Time to Exploit	20 Minutes
Network	Base Mainnet
Novelty of Attack Vector	Rediscovered by Predator

5.2. How Predator Did It

TestMachine's AI-driven smart contract auditing system, Predator, successfully replicated the Socket hack, demonstrating its advanced capabilities in identifying and exploiting complex vulnerabilities across multiple contract layers:

1. **Navigation of complex contract interactions**: The Socket exploit involved multiple layers of contracts, presenting a vast space of potential transactions. Despite this complexity, Predator was able to navigate through these layers, identifying the correct sequence of interactions necessary to execute the exploit.
2. **Precise calldata construction**: Predator leveraged its extensive library of available heuristics and methods to construct the exact calldata required for the exploit. This library, encoded as options for Predator to supply, allowed it to identify and sequence the correct methods and parameters across three layered function calls within a single transaction.
3. **Multi-step transaction sequencing**: The exploit required the precise execution of three-layered function calls within the same transaction. Predator's ability to identify and correctly sequence these calls demonstrates its capacity to understand and manipulate complex smart contract interactions.
4. **Efficient state space exploration**: Despite the large space of potential transactions, Predator's reinforcement learning algorithms efficiently explored this space to identify the correct exploit path. This efficiency is crucial when dealing with complex, multi-contract systems where the number of possible interaction sequences can be astronomical.
5. **Vulnerability detection across contract boundaries**: Predator successfully identified a vulnerability that spanned multiple contracts, including the newly introduced router contract and the existing gateway contract. This cross-contract vulnerability detection is particularly valuable in identifying flaws in complex, interconnected systems like cross-chain bridges.
6. **Privilege escalation detection**: Predator recognized the potential for privilege escalation through the unchecked data field in the `delegatecall`, demonstrating its ability to identify subtle yet critical security flaws related to contract permissions and execution contexts.
7. **Real-time exploit success feedback**: The graph accompanying the description shows a spike in Predator's reward function, indicating the moment when the correct transaction sequence was identified. This real-time feedback mechanism allows Predator to rapidly iterate and refine its exploit strategies.
8. **Simulation of token draining:** In replicating the hack, Predator simulated the draining of tokens from user wallets, mimicking the real-world impact of the vulnerability and providing valuable insights into the potential scale of the exploit.

The reward function associated with this attack is illustrated in Figure 1.

Predator demonstrated its ability to identify and exploit sophisticated vulnerabilities in complex, multi-contract systems like cross-chain bridges. This case highlights the importance of comprehensive, AI-assisted security auditing that can navigate intricate contract interactions and identify vulnerabilities that may be overlooked by traditional auditing methods. The speed and precision with which Predator identified the correct exploit sequence underscore its potential to enhance blockchain security, particularly in the rapidly evolving and high-stakes domain of cross-chain interoperability.

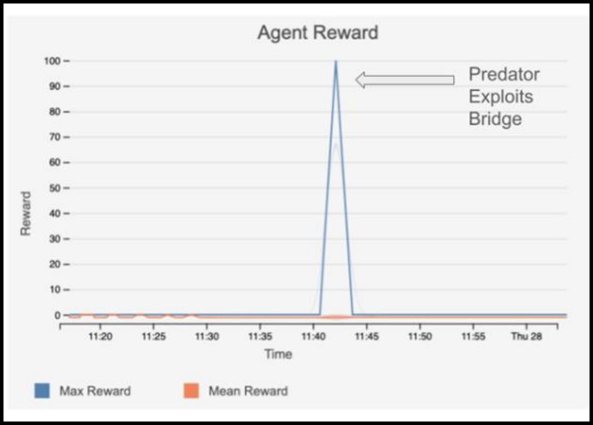

Figure 1: The graph to the right shows the spike in Predator's reward function, a signal that is provided back to the agent in return for its actions, that occurs as Predator identifies the correct transaction sequence.

6. / Rio

6.1. Context

Predator was able to engineer an economic exploit on Rio, a restaking platform built on Eigenlayer, in a testnet environment. This incident is particularly noteworthy as the platform falls within the realm of Real-World Asset (RWA) platforms, which aim to bridge the gap between traditional finance and the crypto economy.

Rio, like its parent project Realio, seeks to democratize access to institutional-grade investments and real-world assets, particularly in the real estate sector. The platform aims to address several key challenges in traditional finance, including high barriers to entry, lack of transparency, liquidity issues, intermediary dependencies, and document tampering risks. By leveraging blockchain technology, Rio intends to make fractional ownership of previously inaccessible assets possible for smaller investors.

The platform's vision aligns with the broader trend of tokenizing real-world assets, aiming to make high-quality investment opportunities globally accessible while enhancing transaction efficiency, transparency, and decentralization.

Table 4: Details of Rio exploit.

Vulnerability	Unanticipated Arbitrage Opportunity
Time to Exploit	9 Hours
Network	Ethereum Testnet (Goerli)
Novelty of Attack Vector	Novel; Discovered by Predator

6.2. How Predator Did It

In this case, TestMachine's AI-driven smart contract auditing system, Predator, demonstrated its ability to identify and exploit not just logical vulnerabilities, but also complex economic imbalances within the protocol. This exploit highlights Predator's advanced capabilities in understanding and manipulating the intricate tokenomics of decentralized finance (DeFi) systems:

1. **Economic modeling and analysis**: Predator showcased its ability to model and analyze the complex economic relationships between different tokens within the Rio ecosystem. This involved understanding the algorithmic links between various staked assets and their impact on token prices.

2. **Identification of arbitrage opportunities**: Rather than exploiting a traditional code vulnerability, Predator identified an unanticipated arbitrage opportunity within the system. This demonstrates the system's capacity to recognize economic inefficiencies that may not be apparent through standard code audits.
3. **Understanding of complex token relationships**: Predator recognized and leveraged the algorithmic exchange rates between different tokens in the ecosystem (`1WETH:1stETH:~0.8WstETH:1RestakedEth`). This understanding of token relationships is crucial for identifying economic vulnerabilities in complex DeFi systems.
4. **Dynamic calculation of optimal exploit parameters**: The agent dynamically computed the optimal amount of Restaked ETH to burn for WETH and WstETH at runtime. This showcases Predator's ability to perform real-time economic calculations to maximize the effectiveness of the exploit.
5. **Liquidity pool analysis**: Predator identified weaknesses in the testnet's liquidity pool configuration, demonstrating its ability to analyze and exploit imbalances in decentralized exchange mechanisms.
6. **Multi-token interaction modeling**: The exploit involved interactions between multiple token types (Restaked ETH, WETH, WstETH), highlighting Predator's capability to model and exploit complex multi-token ecosystems.
7. **Testnet environment exploitation**: By successfully exploiting the testnet environment, Predator demonstrated its ability to identify vulnerabilities in pre-production systems, potentially preventing more severe exploits in live environments.
8. **Economic impact simulation**: In replicating the hack, Predator simulated the economic impact of the arbitrage opportunity, providing valuable insights into potential real-world consequences of such exploits.

In engineering the Rio hack, Predator demonstrated its advanced capabilities in identifying and exploiting economic vulnerabilities in complex DeFi ecosystems. This case highlights the importance of comprehensive, AI-assisted security auditing that goes beyond traditional code analysis to include sophisticated economic modeling and arbitrage detection. The ability to identify such unanticipated arbitrage opportunities underscores Predator's potential to enhance the security and stability of DeFi protocols — in this case in the emerging field of tokenized real-world assets. This exploit serves as a crucial reminder that security in DeFi extends beyond code integrity to include the careful design and balance of economic incentives within the system.

7. / Coinbase

Coinbase

7.1. Context

Coinbase, one of the leading cryptocurrency exchanges, engaged TestMachine in a proactive security assessment of token contracts prior to deployment on their exchange. As part of this rigorous evaluation process, the Coinbase team deliberately injected vulnerabilities into their smart contract code to test the capabilities of TestMachine's Predator auditing system.

The specific vulnerability in question was related to Role-Based Access Control (RBAC), a crucial security mechanism in smart contracts that manages permissions for different functions. Unbeknownst to TestMachine, a vulnerability was introduced that granted overly permissive access to an admin function, potentially allowing unauthorized users to execute privileged operations.

Table 5: Details of Coinbase token exploit.

Vulnerability	Role Base Access Control (RBAC)
Time to Exploit	5 Minutes
Network	Pre-Deploy
Novelty of Attack Vector	Novel; Discovered by Predator

7.2. How Predator Did It

TestMachine's AI-driven smart contract auditing system, Predator, demonstrated its advanced capabilities in rapidly identifying and exploiting the deliberately injected vulnerability:

1. **Rapid vulnerability detection**: Predator identified the RBAC vulnerability within minutes, showcasing its efficiency in analyzing complex smart contract structures and access control mechanisms.
2. **Invariant-based testing**: The system utilized a specifically crafted invariant designed to verify the access-control mechanisms of the Smart Wallet. This approach highlights Predator's ability to leverage custom-defined security properties to guide its analysis.
3. **Function modifier analysis**: Predator successfully identified that the vulnerability stemmed from an overly permissive function modifier. This demonstrates its capability to scrutinize fine-grained details of smart contract implementations, including subtle issues in access control logic.

4. **Automated exploit generation**: Upon identifying the vulnerability, Predator autonomously generated the necessary transactions to exploit the flaw, simulating how a malicious actor might take advantage of the overly permissive access control.
5. **Comprehensive RBAC evaluation**: The system's ability to detect this specific vulnerability indicates its broader capability to evaluate complex RBAC structures, a critical aspect of smart contract security.
6. **Simulated attack execution**: In the controlled environment, Predator simulated the execution of unauthorized admin functions, demonstrating the potential real-world impact of the vulnerability.
7. **False positive minimization**: By correctly identifying a deliberately injected vulnerability, Predator demonstrated its ability to minimize false positives, a crucial factor in practical security auditing.
8. **Adaptability to diverse contract structures**: The successful identification of this vulnerability in Coinbase's token contracts showcases Predator's adaptability to various smart contract architectures and implementations.
9. **Real-time feedback mechanism**: Predator's ability to quickly signal the failure of the access-control mechanism through its invariant-based approach highlights its effectiveness in providing real-time feedback during the auditing process.
10. **Collaboration with industry leaders**: This case demonstrates Predator's capability to work effectively with leading cryptocurrency platforms like Coinbase, adapting to their specific security requirements and testing methodologies.

By successfully identifying the deliberately injected RBAC vulnerability in Coinbase's token contracts, Predator demonstrated its effectiveness as a cutting-edge tool for smart contract security auditing. This case underscores the importance of sophisticated, AI-driven security analysis in the rapidly evolving cryptocurrency ecosystem. The speed and accuracy with which Predator identified the vulnerability highlight its potential to enhance the security of token contracts before they are deployed on major exchanges, thereby protecting users and maintaining the integrity of the cryptocurrency market.

Moreover, this successful collaboration with Coinbase showcases the growing recognition of AI-driven tools like Predator in the blockchain industry. It emphasizes the shift towards more proactive and comprehensive security measures, where potential vulnerabilities are identified and addressed before they can be exploited in live environments. This approach not only enhances the security of individual token contracts but also contributes to the overall robustness and trustworthiness of the cryptocurrency ecosystem.