A07: Smart Contract Audit Challenge

Find the Bugs Before the Hackers Do

Prof. Joerg Osterrieder

Spring 2026

Assignment Overview

Learning Objectives

- Identify common smart contract vulnerabilities
- Design attack scenarios that exploit vulnerabilities
- Propose concrete fixes for each vulnerability
- Analyze the economic impact of smart contract exploits

Assignment Details

- Time: 60 minutes
- Format: Groups of 2–3 students
- Difficulty: Hard
- Points: 50 (+10 bonus)

Grading Breakdown

- Completed worksheet: 30 pts
- Group presentation: 15 pts
- Class participation: 5 pts
- Bonus (advanced contracts): +10 pts

A07:

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

Step 1: Contract Review

20 min — 30 pts (worksheet)

Steps

1. Receive your 3 vulnerable contract handouts
2. Use the vulnerability reference guide as your checklist
3. For each contract, identify:
   - The vulnerability type
   - The vulnerable function or line
   - Why it's dangerous
4. Record findings on the audit worksheet

Materials

- 3 contract code handouts
- Vulnerability reference guide
- Audit/attack worksheet

Common Vulnerability Types

- Reentrancy (calling external contracts before updating state)
- Integer overflow/underflow
- Access control (missing `onlyOwner` or similar)
- Unchecked return values

A07:

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

Step 2: Attack Development

15 min

Steps

1. For each vulnerability found, design an attack scenario
2. Describe step by step how an attacker would exploit it
3. Estimate the economic impact (how much could be stolen/lost?)
4. Propose a specific code fix for each vulnerability

Think Like an Attacker

"A rational attacker considers: (1) How much can I gain? (2) What does it cost me? (3) Can I be caught? If profit > cost, someone will try it."

Fix Categories

- Checks-Effects-Interactions pattern (reentrancy)
- SafeMath or Solidity 0.8+ (overflow)
- Access control modifiers (authorization)
- Require statements for return values

A07:

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

Step 3: Analysis & Comparison

10 min

Steps

1. Rank your 3 vulnerabilities by severity (Critical / High / Medium / Low)
2. Which vulnerabilities could be combined for a compound attack?
3. As a rational attacker with limited resources, which vulnerability would you exploit first and why?
4. Discuss: which fix is most urgent?

Severity Criteria

- **Critical**: Direct loss of all funds
- **High**: Significant fund loss or contract takeover
- **Medium**: Limited fund loss or DoS
- **Low**: Informational or minor impact

Compound Attacks

"Real-world exploits often chain multiple vulnerabilities. The DAO hack combined reentrancy with a recursive call pattern."

A07:

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

Step 4: Presentation

15 min total (5 min/group)

Presentation Structure

1. Vulnerability explanation: describe what you found and why it's dangerous
2. Attack demonstration: walk through your attack scenario step by step
3. Fix proposal: present your recommended code fix
4. Economic analysis: estimate the financial impact

Bonus Challenge (+10 pts)

- 3 additional advanced contracts:
  - Flash loan attack patterns
  - Oracle manipulation vulnerabilities
- Worth up to +10 bonus points
- Attempt only after completing the core 3 contracts

Deliverables

- Completed audit worksheet (all 3 contracts)
- Group presentation (5 min)
- Active class participation

A07:

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

Real-World Smart Contract Exploits

The DAO Hack (2016)

- Vulnerability: Reentrancy
- Impact: $60 million stolen
- Consequence: Ethereum hard fork (ETH/ETC split)
- Lesson: Always update state before external calls

Poly Network (2021)

- Vulnerability: Access control flaw
- Impact: $600 million stolen (returned by hacker)
- Consequence: Largest DeFi hack at the time
- Lesson: Cross-chain bridges are high-risk targets

Billions Lost

"DeFi hacks have caused billions in losses. Smart contract auditing is one of the most in-demand skills in blockchain."

A07:

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

Audit Tips

Audit Checklist

- Check all external calls (reentrancy risk)
- Check all math operations (overflow risk)
- Check access control on sensitive functions
- Check that return values are verified
- Look for state changes after external calls

Professional Audit Process

- Automated tools: Slither, Mythril, Echidna
- Manual review: line-by-line code inspection
- Formal verification: mathematical proofs
- Bug bounties: incentivize white-hat hackers

Key Principle

"The most dangerous bugs are the ones that look correct. Always ask: "What if a malicious user called this function in an unexpected way?"

A07:

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

Get Started

Assignment Page

digital-ai-finance.github.io/crypto-economics/assignments/A07_audit_challenge/instructions.html

All Assignments

digital-ai-finance.github.io/crypto-economics/assignments/index.html

Grab your contract handouts and start hunting for bugs!

You have 60 minutes: 20 min review, 15 min attack design, 10 min analysis, 15 min presentations.