L42: Flash Loans and Composability

Module F: Advanced Topics

Blockchain & Cryptocurrency Course

December 2025

What is a Flash Loan?

- **Definition**: Uncollateralized loan that must be borrowed and repaid within a single transaction
- **Key Property**: *Atomicity* – loan and repayment are atomic (all-or-nothing)
- **If repayment fails**: Entire transaction reverts, lender loses nothing
- **No Collateral Required**: Enabled by smart contract execution model
- **Loan Size**: Unlimited (constrained only by liquidity pool)
- **Duration**: Fraction of a second (one block)
- **Unique to DeFi**: Impossible in traditional finance

Traditional Loans vs Flash Loans

Property	Traditional Loan	Flash Loan
Collateral	Required (often ¿100%)	None
Duration	Days/months/years	Single transaction (seconds)
Creditworthiness	Required (KYC, credit score)	Not required
Repayment Guarantee	Legal contracts	Smart contract atomicity
Risk to Lender	Default risk	Zero (transaction reverts)
Use Cases	Consumption, investment	Arbitrage, refinancing

Paradigm Shift: Code execution guarantees replace legal enforcement

Flash Loan Mechanism

1. **Borrower calls flash loan function** on lending protocol (e.g., Aave)
2. **Protocol transfers tokens** to borrower's contract
3. **Borrower's contract executes arbitrary logic**:
   - Trade on DEXs
   - Refinance debt positions
   - Liquidate collateral
   - Exploit arbitrage opportunities
4. **Borrower repays loan + fee** (typically 0.05-0.09%)
5. **Protocol checks repayment**:
   - If successful: Transaction confirmed
   - If failed: `revert()` – entire transaction cancelled

Flash Loan Code Example (Simplified)

```
// Borrower's contract
function executeFlashLoan() external {
    ILendingPool(aave).flashLoan(
        address(this),        // receiver
        DAI_ADDRESS,         // asset to borrow
        1000000 ether,       // amount (1M DAI)
        ""                   // params
    );
}

function executeOperation(
    address asset,
    uint256 amount,
    uint256 premium,
    address initiator
) external returns (bool) {
    // 1. Received 1M DAI
    // 2. Execute arbitrage/liquidation/refinancing
    doArbitrage();
    // 3. Approve repayment
    IERC20(asset).approve(msg.sender, amount + premium);
    return true;  // Success: repay. Failure: revert entire transaction
}
```

DeFi Composability

- **Composability**: DeFi protocols are like “money legos”
- **Permissionless Integration**: Any contract can call any other contract
- **Atomic Transactions**: Multiple protocol interactions in one transaction
- **Examples of Composability**:
  1. Swap on Uniswap → Deposit into Aave → Borrow against collateral
  2. Flash loan → Liquidate position → Swap collateral → Repay
  3. Borrow from Compound → Yield farm on Curve → Stake LP tokens
- **Innovation Engine**: New protocols build on existing primitives
- **Risk**: Cascading failures, attack surface expansion

Flash Loan Use Case 1: Arbitrage

- **Scenario**: ETH trades at $2000 on Uniswap, $2020 on SushiSwap
- **Traditional Arbitrage**: Requires capital upfront
- **Flash Loan Arbitrage**:
  1. Borrow 1000 ETH via flash loan
  2. Buy 1000 ETH on Uniswap ($2,000,000)
  3. Sell 1000 ETH on SushiSwap ($2,020,000)
  4. Repay flash loan + fee ($2,000,000 + $1,000)
  5. Profit: $19,000 in one transaction
- **Capital Required**: Only gas fees ($50-$200)
- **Market Impact**: Arbitrage opportunities closed instantly
- **Democratization**: Anyone can be arbitrageur, not just whales

Flash Loan Use Case 2: Collateral Swap

- **Problem**: User has debt collateralized with Asset A, wants to switch to Asset B
- **Traditional Solution**: Close position, pay interest, open new position (costly)
- **Flash Loan Solution**:
  1. Borrow Asset A via flash loan
  2. Repay existing debt on Protocol 1
  3. Withdraw original collateral
  4. Deposit new collateral (Asset B)
  5. Borrow Asset A on Protocol 2
  6. Repay flash loan
- **Result**: Collateral swapped in one transaction, zero liquidation risk
- **Fee**: Only flash loan fee (0.05-0.09%)

Flash Loan Use Case 3: Self-Liquidation

- **Scenario**: User's collateralized position near liquidation threshold
- **Problem**: External liquidation incurs 5-15% penalty
- **Flash Loan Self-Liquidation**:
  1. Borrow funds via flash loan
  2. Repay own debt
  3. Withdraw collateral
  4. Sell collateral to repay flash loan
- **Savings**: Avoid liquidation penalty, keep liquidation bonus
- **Example**: Save 10% penalty on \$100,000 position = \$10,000 saved

Flash Loan Providers

Protocol	Fee	Assets	Max Liquidity
Aave	0.09%	30+ tokens	$10B+
dYdX	0%	ETH, USDC, DAI	$500M
Uniswap V3	Variable	Any pool token	Pool-dependent
Balancer	0.05%	Pool tokens	Pool-dependent

- **Aave**: Largest provider, most liquid
- **dYdX**: Free flash loans (margin trading focused)
- **Uniswap V3**: Flash swaps (borrow any token in pool)

Flash Loan Attacks: Overview

- **Dark Side**: Flash loans enable large-scale attacks with zero capital
- **Attack Pattern**:
  1. Borrow massive amount via flash loan
  2. Manipulate protocol state (price oracle, governance, liquidity)
  3. Exploit manipulation for profit
  4. Repay flash loan
- **Impact**: $500M+ stolen via flash loan attacks (2020-2023)
- **Key Insight**: Flash loans amplify existing vulnerabilities
- **Not the flash loan's fault**: Underlying protocol weakness is the root cause

Flash Loan Attack Example: Oracle Manipulation

- **Vulnerable Protocol**: Uses on-chain DEX price as oracle (e.g., single Uniswap pool)
- **Attack Steps**:
  1. Borrow 10,000 ETH via flash loan
  2. Buy all TOKEN on Uniswap pool (manipulate price 10x higher)
  3. Protocol oracle reads inflated price
  4. Borrow stablecoins using overvalued TOKEN as collateral
  5. Sell TOKEN back to pool (price normalizes)
  6. Repay flash loan + keep borrowed stablecoins
- **Real Example**: Harvest Finance attack (Oct 2020) – $34M stolen
- **Mitigation**: Use time-weighted average price (TWAP) or Chainlink oracles

Flash Loan Attack Example: Governance Attack

- **Vulnerable Protocol**: Snapshot-based governance (voting power = token balance at block)
- **Attack Steps**:
  1. Borrow 10M governance tokens via flash loan
  2. Vote on malicious proposal in same transaction
  3. Repay flash loan
- **Malicious Proposals**:
  - Change protocol parameters (fees, interest rates)
  - Drain treasury
  - Upgrade contract to backdoored version
- **Real Example**: BZx Protocol (Feb 2020) – governance manipulation
- **Mitigation**: Time-delayed voting, vote locking, delegation mechanisms

Notable Flash Loan Attacks

- **BZx (Feb 2020)**: $350k stolen via oracle manipulation
- **Harvest Finance (Oct 2020)**: $34M via USDC/USDT pool manipulation
- **Cream Finance (Oct 2021)**: $130M via reentrancy + flash loan
- **Beanstalk (Apr 2022)**: $182M via governance takeover (flash loan borrowed governance tokens)
- **Mango Markets (Oct 2022)**: $110M via oracle manipulation (borrowed $116M USDC to inflate MNGO price)
- **Euler Finance (Mar 2023)**: $200M via donation attack + flash loan

- **Common Thread**: Flash loans exploit existing vulnerabilities at scale

Defense Mechanisms

1. **Decentralized Oracles**: Use Chainlink, Band Protocol (not single DEX)
2. **Time-Weighted Average Price (TWAP)**: Average price over multiple blocks
3. **Reentrancy Guards**: Prevent recursive contract calls
4. **Governance Delays**: Timelock on parameter changes (24-48h)
5. **Vote Locking**: Require tokens locked for X days before voting
6. **Flash Loan Detection**: Check if `tx.origin == msg.sender`
7. **Circuit Breakers**: Pause protocol if anomalous activity detected
8. **Liquidity Caps**: Limit max borrow amount per transaction

Economic Impact of Flash Loans

Positive Effects

- Democratize arbitrage (no capital barrier)
- Increase market efficiency
- Enable capital-efficient refinancing
- Innovation in DeFi tooling
- Liquidation bots improve protocol health

Negative Effects

- Enable zero-capital attacks
- Amplify protocol vulnerabilities
- MEV extraction (miner extractable value)
- Governance manipulation risk
- Increased attack surface

Net Assessment: Powerful tool that magnifies both good and bad aspects of protocol design

MEV and Flash Loans

- **MEV (Maximal Extractable Value)**: Profit from transaction ordering
- **Flash Loans + MEV**: Amplify arbitrage and liquidation profits
- **Searchers**: Bots scanning mempool for MEV opportunities
- **Techniques**:
  - **Front-running**: Place transaction before victim's transaction
  - **Back-running**: Place transaction after victim's transaction
  - **Sandwich Attacks**: Front-run + back-run (manipulate price around victim)
- **Flashbots**: Democratize MEV extraction, reduce gas wars
- **MEV Volume**: $600M+ extracted in 2023

Building a Flash Loan Strategy

1. **Identify Opportunity**:
   - Monitor price discrepancies across DEXs
   - Track liquidatable positions
   - Analyze governance proposals
2. **Design Atomic Transaction**:
   - Flash loan → Execute strategy → Repay
   - Account for fees, slippage, gas costs
3. **Implement Smart Contract**:
   - Integrate with lending protocol (Aave, dYdX)
   - Add safety checks (minimum profit threshold)
4. **Simulate and Test**:
   - Use Hardhat/Foundry for local testing
   - Fork mainnet for realistic simulation
5. **Deploy and Monitor**:
   - Gas optimization critical (failed transactions cost gas)

Flash Loans and Regulation

- **Legal Gray Area**: No clear regulatory framework
- **Questions**:
  - Are flash loan attacks theft or exploitation of code?
  - Is the protocol or attacker liable?
  - Code is law vs legal enforcement
- **Precedents**:
  - Mango Markets attacker Avi Eisenberg arrested (Oct 2022)
  - Charged with market manipulation, not flash loan use
- **Regulatory Uncertainty**: Chilling effect on DeFi innovation
- **Protocol Responsibility**: Bug bounties, audits, insurance

Future of Flash Loans

- **Cross-Chain Flash Loans**: Borrow on Ethereum, use on Arbitrum/Polygon
- **Flash Minting**: Mint unbacked stablecoins within transaction (e.g., MakerDAO)
- **Flash Loan Aggregators**: Optimize across multiple providers
- **Improved Defenses**: Protocol-level flash loan resistance
- **Institutional Adoption**: Hedge funds using flash loans for arbitrage
- **Layer 2 Integration**: Cheaper gas makes smaller arbitrage profitable
- **Regulatory Clarity**: Potential restrictions on governance voting with borrowed tokens

Summary

- **Flash loans**: Uncollateralized loans repaid in single atomic transaction
- **Enabled by smart contract atomicity**: Revert on failure = zero lender risk
- **Use cases**: Arbitrage, collateral swaps, self-liquidation, refinancing
- **Composability**: DeFi protocols as interoperable building blocks
- **Attacks**: Oracle manipulation, governance takeover ($500M+ stolen)
- **Not inherently malicious**: Flash loans amplify existing protocol vulnerabilities
- **Defenses**: Decentralized oracles, TWAP, vote locking, circuit breakers
- **Impact**: Democratize capital-intensive strategies, increase market efficiency
- **Future**: Cross-chain flash loans, better protocol defenses, regulatory clarity