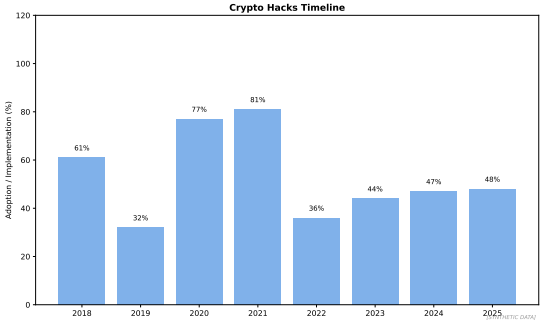
Lesson 23: Security and Hacks

Module 2: Blockchain Fundamentals

Digital Finance

The Stakes: Billions Lost to Hacks

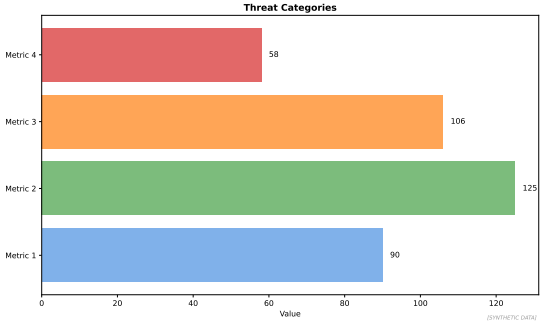


Major Incidents:

- **2016:** The DAO ($60M) – reentrancy
- **2021:** Poly Network ($611M) – access control (returned)
- **2022:** Ronin Bridge ($625M) – compromised keys
- **2022:** Wormhole Bridge ($325M) – signature verification
- **Total (2020–2024):** >$10 billion stolen

Key concepts from this slide inform practical applications in finance.

Threat Landscape



Attack Vectors:

- **Smart Contract Bugs:** Logic errors, reentrancy, overflow
- **Bridge Vulnerabilities:** Cross-chain security weaknesses
- **Private Key Compromise:** Phishing, social engineering, malware
- **Oracle Manipulation:** Flash loan attacks on price feeds
- **Governance Attacks:** Token-based voting exploits

Key concepts from this slide inform practical applications in finance.

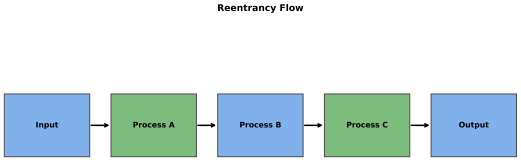
Reentrancy: The DAO Hack (2016)

Vulnerability:

- Contract sends ETH before updating balance
- Recipient contract calls back
- Recursive withdrawals drain funds

Impact:

- 3.6M ETH stolen ($60M)
- Led to Ethereum hard fork
- ETH/ETC split

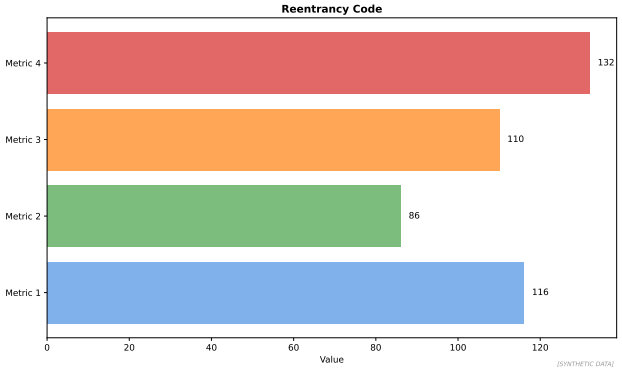


[SYNTHETIC DATA]

Prevention: Checks-Effects-Interactions pattern, reentrancy guards (OpenZeppelin)

Key concepts from this slide inform practical applications in finance.

Reentrancy Code Example



Fix: Checks-Effects-Interactions

- **Checks:** Validate conditions (sufficient balance)
- **Effects:** Update state (subtract balance)
- **Interactions:** External calls (send ETH)

Modern Protection: OpenZeppelin `ReentrancyGuard` modifier

Case studies provide concrete evidence of technology impact and adoption patterns.

Integer Overflow/Underflow

Problem (Pre-Solidity 0.8):

- uint256 max: $2^{256} - 1$
- Overflow: max + 1 wraps to 0
- Underflow: 0 - 1 wraps to max
- Silent failures (no revert)

Example Attack:

- Balance: 100 tokens
- Transfer 200 (should fail)
- `balance - 200` underflows to huge number
- Exploit unlimited tokens

Overflow Visualization

[SYNTHETIC DATA]

Fix: Solidity 0.8+ has automatic overflow checks, or use SafeMath library

Understanding the process flow is key to identifying optimization opportunities.

Access Control Vulnerabilities

Common Mistakes:

- Missing `onlyOwner` modifiers on critical functions
- Default function visibility (public vs internal)
- Incorrect permission checks

Example: Parity Wallet Hack (2017)

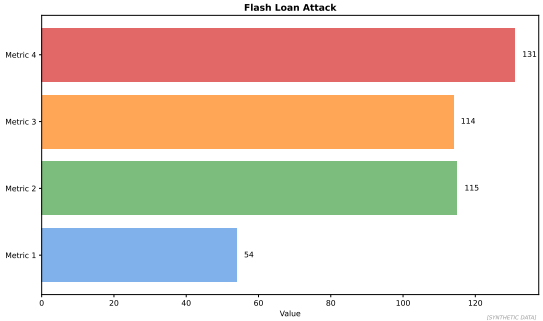
- Multi-sig library had unprotected `initWallet()` function
- Anyone could call and become owner
- Attacker initialized, then called `kill()` (selfdestruct)
- $300M frozen permanently

Best Practice:

- Explicitly declare visibility (public, external, internal, private)
- Use OpenZeppelin `Ownable` and `AccessControl`
- Audit all privileged functions

Key concepts from this slide inform practical applications in finance.

Oracle Manipulation: Flash Loan Attacks



Attack Pattern (e.g., Harvest Finance, $24M):

1. Flash loan large amount of Token A
2. Swap on low-liquidity DEX, manipulate price
3. Protocol using DEX price oracle now sees inflated price
4. Exploit: borrow over-collateralized or dump tokens at fake price
5. Repay flash loan, profit

Security analysis identifies vulnerabilities and helps design robust systems.

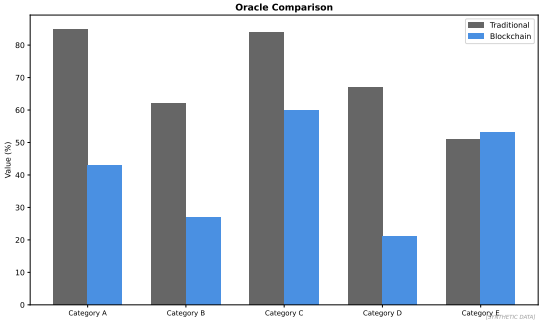
Oracle Security: Defense Mechanisms

Vulnerable:

- Single DEX as price source
- Spot price (current block)
- Low liquidity pools

Secure:

- Time-Weighted Average Price (TWAP)
- Multiple oracle sources (Chainlink)
- High liquidity requirements
- Price deviation limits



Chainlink: Decentralized oracle network, aggregates data from multiple providers

Security analysis identifies vulnerabilities and helps design robust systems.

Bridge Hacks: Cross-Chain Vulnerabilities

Why Bridges are Risky:

- Centralized honeypots (large TVL)
- Complex multi-sig schemes
- Cross-chain validation challenges
- Centralized validators

Attack Types:

- Compromised validator keys
- Signature verification bugs
- Replay attacks

Bridge Architecture

[SYNTHETIC DATA]

AI and ML are transforming financial services through automation and prediction.

Ronin Bridge Hack (2022): $625M Stolen

Context:

- Ronin: Ethereum sidechain for Axie Infinity game
- Bridge: Transfer assets between Ethereum and Ronin
- Requires 5-of-9 validator signatures for withdrawals

Attack:

1. Attacker compromised 4 Sky Mavis (developer) validator keys (spear phishing)
2. Exploited backdoor in Axie DAO validator (5th signature)
3. Forged withdrawal transaction: 173,600 ETH + 25.5M USDC
4. Took 6 days to detect (low monitoring)

Root Cause: Centralization (5 of 9 validators controlled by single entity), poor key management

---

Key concepts from this slide inform practical applications in finance.

Wormhole Bridge Hack (2022): $325M

Mechanism:

- Wormhole: Bridge between Ethereum and Solana
- Guardians validate cross-chain messages

Vulnerability:

- Signature verification function had bug
- Attacker bypassed guardian signature check
- Minted 120,000 wrapped ETH on Solana without depositing on Ethereum

Outcome:

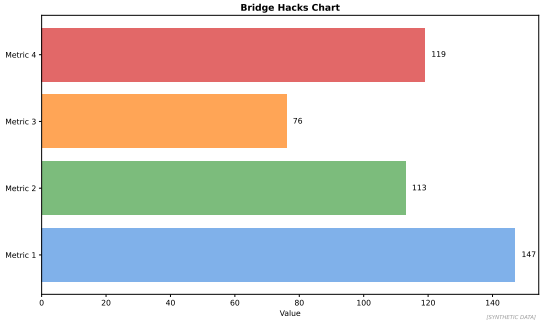
- Attacker bridged fake wETH to Ethereum, sold for real ETH
- Jump Crypto (parent company) refunded users ($325M bailout)

Lesson: Even well-funded projects have critical bugs, audits not foolproof

---

Key concepts from this slide inform practical applications in finance.

Bridge Hacks: Total Damage



Major Bridge Hacks (2021–2022):

- Ronin: $625M, Poly Network: $611M, Wormhole: $325M, Nomad: $190M, Harmony: $100M
- Total: >$2 billion (over 50% of all DeFi hacks)

Trend: Bridges are highest-value targets (centralized trust, large TVL)

Key concepts from this slide inform practical applications in finance.

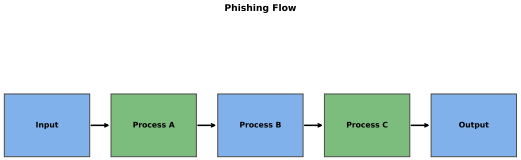
Phishing and Social Engineering

Common Attacks:

- Fake wallet extensions (clipboard hijack)
- Malicious dApp frontends
- Discord/Telegram impersonation
- Fake support ("verify your wallet")
- Address poisoning

Example:

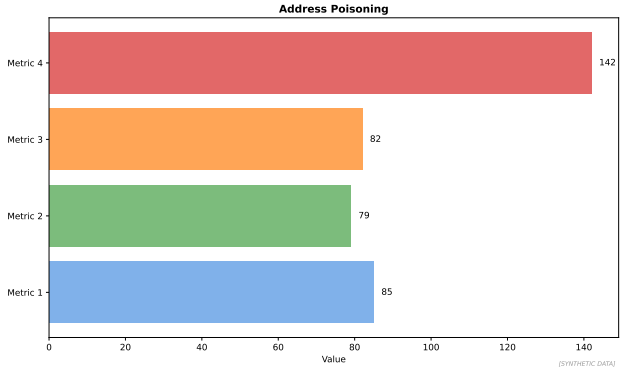
- User clicks malicious link
- Connects wallet to fake dApp
- Signs `setApprovalForAll` transaction
- Attacker drains NFTs/tokens



[SYNTHETIC DATA]

Key concepts from this slide inform practical applications in finance.

Address Poisoning Attack



Attack:

1. Attacker creates vanity address matching first/last chars of victim's address
2. Sends 0 ETH transaction to victim (appears in history)
3. Victim copies address from history (sees first/last match, assumes correct)
4. Sends funds to attacker's address

Prevention: Verify entire address, use address book, hardware wallet confirmation screens

Security analysis identifies vulnerabilities and helps design robust systems.

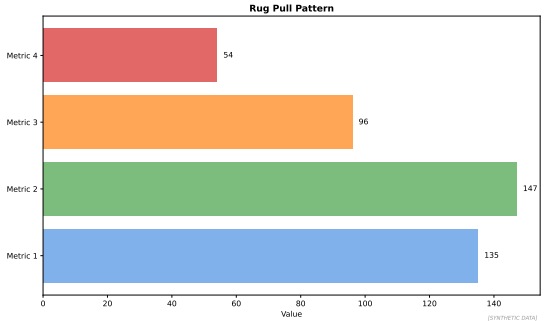
Rug Pulls: Exit Scams

Types:

- **Liquidity Removal:** Dev drains LP, token worthless
- **Backdoor Functions:** Hidden mint/burn/pause
- **Honeypot:** Can buy but not sell

Example: Squid Game Token (2021)

- Market cap: $3M in days
- Could buy, not sell (code restriction)
- Developers cashed out, disappeared



Red Flags: Anonymous teams, no audit, centralized control, unrealistic promises

Key concepts from this slide inform practical applications in finance.

Governance Attacks

Scenario:

- Protocol governed by token holders (1 token = 1 vote)
- Attacker acquires majority via flash loan or market buy
- Proposes malicious governance action (drain treasury)
- Votes pass, execute attack, repay flash loan

Example: Beanstalk (2022, $182M)

- Attacker took flash loan of governance tokens
- Voted to approve malicious proposal
- Drained $182M from protocol
- Repaid flash loan in same transaction

Defense: Time-locks on proposals, vote escrow (lock tokens for voting), quorum requirements

---

Security analysis identifies vulnerabilities and helps design robust systems.

Auditing: Not a Silver Bullet

Audit Process:

- Security firm reviews smart contract code
- Identifies vulnerabilities, best practice violations
- Provides report with severity levels (critical, high, medium, low)
- Cost: \$50K–\$500K+ depending on complexity

Limitations:

- Audits are snapshots (code changes after audit)
- Cannot find all bugs (Wormhole was audited)
- Economic attacks (flash loans, governance) hard to model
- Audit shopping (only publish favorable audits)

Best Practice: Multiple audits (Trail of Bits, OpenZeppelin, Consensys Diligence), public bug bounties

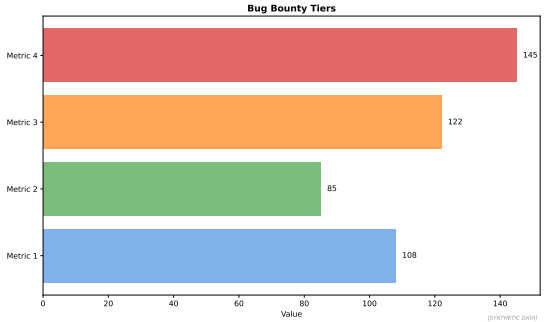
Bug Bounties: Whitehat Incentives

Programs:

- Pay security researchers for finding bugs
- Tiered rewards (critical: $1M+)
- Platforms: Immunefi, HackerOne

Examples:

- Aurora: $6M bounty (critical)
- Wormhole: $10M (post-hack)
- MakerDAO: $10M max



Whitehats Saved: >$50M in 2023 via responsible disclosure

Key concepts from this slide inform practical applications in finance.

Security Best Practices

For Developers:

- Use audited libraries (OpenZeppelin)
- Multiple audits before mainnet launch
- Time-locks on upgrades and governance
- Comprehensive testing (unit, integration, fuzzing)
- Bug bounty programs
- Incident response plan

For Users:

- Hardware wallets for large amounts
- Verify contract addresses (bookmarks, not Google)
- Revoke token approvals regularly (revoke.cash)
- Never share seed phrases
- Be skeptical of high yields (if too good to be true...)

Security analysis identifies vulnerabilities and helps design robust systems.

Summary

- **Reentrancy:** External calls before state updates, use checks-effects-interactions
- **Overflow/Underflow:** Solidity 0.8+ auto-checks, or SafeMath
- **Oracle Manipulation:** Use TWAP, Chainlink, avoid single DEX price
- **Bridge Hacks:** Centralized validators, key compromise ($2B+ stolen)
- **Phishing:** Social engineering, verify addresses, revoke approvals
- **Best Practices:** Audits, bug bounties, time-locks, user vigilance

Next Lesson: Regulation and future – MiCA, SEC, real-world assets, careers