7 Phases of smart contract hacking

Martin Abbatemarco
@tinchoabbate
Martin Abbatemarco
Security Researcher @ OpenZeppelin
Systems Engineer, doing master’s degree in Cybersecurity & Cyberdefense
Former pentester at a Big-Four
You’re a pentester / bug hunter / white hat / ...
You’ve heard about “blockchain” and “smart contracts”
Once hailed as unhackable, blockchains are now getting hacked

More and more security holes are appearing in cryptocurrency and blockchain platforms, and as a result, there have been significant financial losses.

You’ve heard about “blockchain” and “smart contracts”
You’ve heard about “blockchain” and “smart contracts”

Once hailed as unhackable, blockchains are now getting hackes. More and more security holes are appearing in cryptos that hackers can take advantage of. The biggest smart contract hacks in history or how to endanger up to US $2.2 billion
You’ve heard about "blockchain" and "smart contracts". Once hailed as unhackable, blockchains are now getting hackable. More and more security holes are appearing in cryptography and code, threatening the future of blockchain and cryptocurrencies. The biggest smart contract hacks in history or how to endanger up to US $2.2 billion that the DAO was all too human.
You've heard about “blockchain” and “smart contracts.”

Once hailed as un hack able, blockchains are now getting hack

SpankChain Loses $40K in Hack Due to Smart Contract Bug

The biggest smart contract hacks in history or how to endanger up to US $2.2 billion

A $50 Million Hack Just Showed That the DAO Was All Too Human
Once hailed as unhackable, blockchains are now getting hack security holes are appearing in cry

SpankChain Loses $40K in Hack Due to Smart Contract Bug

The biggest smart contract hacks in history or how to endanger up to US $2.2 billion

smart contracts”

A Smart Contract’s Secrets Revealed: The Trillion Dollar Problem

⚠️ MakerDAO Critical Vulnerability Notice
Once hailed as unhackable, blockchains are now getting hack security holes are appearing in cry...
Once hailed as unhackable, blockchains are now getting hackable. Security holes are appearing in cryptos.

SpankChain Loses $40K in Hack Due to SpankChain

The biggest smart contract hacks in history or how to endanger up to US $2.2 billion

Argent

High Severity Vulnerability

OpenZeppelin Security Notice
You've heard about "blockchain" and "smart contracts"

SpankChain Loses $40K in Hack Due to Smart Contracts

Another Ethereum DeFi hack: Balancer Pools looses $500,000

Once hailed unhackable,
Million in Ethereum, Tether are now get
How DeFi’s Lendf.me Lost $25
And More: A Breakdown
Could you hack a smart contract in Ethereum?

Is your know-how from other infosec realms of any use?
Could you hack a smart contract in Ethereum?
You do know the “seven” phases of a typical security engagement.
0. scoping
1. recon
2. identification of vulns
3. exploiting
4. post-exploitation
5. report
6. fixes & retest
0. scoping
2606:4700::6811:b055
IP address

0x36B2a6610Fc1a5F441277
Eb30d83Fd3601Ee68a4
Ethereum address

7 phases of Smart Contract Hacking - Phase 0: scoping
2606:4700::6811:b055
IP address

0x36B2a6610Fc1a5F441277
E30d83Fd3601Ee68a4
Ethereum address

many other related addresses

7 phases of Smart Contract Hacking - Phase 0: **scoping**
2606:4700::6811:b055
IP address

cloudflare.com
domain name

0x36B2a6610Fc1a5F441277
Ethereum address

Eb30d83Fd3601Ee68a4

johndoe.eth
domain name

7 phases of Smart Contract Hacking - Phase 0: *scoping*
7 phases of Smart Contract Hacking - Phase 0: **scoping**
Most times you’ll have **Solidity** source code

`github.com/org/project/contracts/*.sol`

or

verified source code in Etherscan
7 phases of Smart Contract Hacking - Phase 0: **scoping**

Verified source code of DAI token contract
If you don’t have access to source code, decompile EVM bytecode
1. reconnaissance

a.k.a. what’s in there?
7 phases of Smart Contract Hacking - Phase 1: reconnaissance
7 phases of Smart Contract Hacking - Phase 1: reconnaissance
7 phases of Smart Contract Hacking - Phase 1: reconnaissance
7 phases of Smart Contract Hacking - Phase 1: reconnaissance
public class MyClass {
    /* Attributes */
    private String text = "Hey";
    /* Methods */
    function String getText() {
        return text;
    }
}

contract MyContract {
    /* State variables */
    string private text = "Hey";
    /* Functions */
    function getText() public returns (string) {
        return text;
    }
}
7 phases of Smart Contract Hacking - Phase 1: reconnaissance
In reconnaissance, we want to understand:

- Architecture
- Inheritance chains
- Interactions between contracts
- Contracts’ exposed API as entry points for attacks
- Integration points with other protocols
- Intention
- Roles
In reconnaissance, we want to understand:

- Architecture
- Inheritance chains
- Interactions between contracts
- Contracts’ exposed API as entry points for attacks
- Integration points with other protocols
- Intention
- Roles

7 phases of Smart Contract Hacking - Phase 1: reconnaissance
In reconnaissance, we want to understand

- **Architecture**
- **Inheritance chains**
  - Interactions between contracts
  - Contracts’ exposed API as entry points for attacks
- **Integration points** with other protocols
- **Intention**
- **Roles**

7 phases of Smart Contract Hacking - Phase 1: *reconnaissance*
Inheritance chains
(Etherscan, Surya, Slither)
7 phases of Smart Contract Hacking - Phase 1: reconnaissance

Exposed functions
(Slither)
IDE plugins
(VSCode Solidity Auditor)

7 phases of Smart Contract Hacking - Phase 1: reconnaissance
7 phases of Smart Contract Hacking - Phase 1: reconnaissance

Your diagrams!
2. vuln identification

a.k.a. omg, look what’s in there!
If you’re a beginner, start with what’s known.

7 phases of Smart Contract Hacking - Phase 2: **vuln identification**
Automated and manual
7 phases of Smart Contract Hacking - Phase 2: \textit{vuln identification}

\begin{itemize}
\item Slither, the Solidity source analyzer
\item Manticore
\item VeriSol (Verifier for Solidity)
\item Echidna: A Fast Smart Contract Fuzzer
\item Pakala
\item Mythril
\item and many more
\end{itemize}
Automated

7 phases of Smart Contract Hacking - Phase 2: *vuln identification*

and *many* more!
Focus on business logic and the **intention**

Special care in **sensitive** functions
- Convoluted and mixed logic
- Deposits / withdrawals of tokens and ETH
- Balance snapshots
- Handling decimals, arrays, signatures
- Dangerous math
- External calls to untrusted contracts
- Low-level calls, use of assembly
Manual

**Interactions** with external dependencies

- Other protocols
- Decentralized exchanges
- Price oracles

**Impact of unexpected / extreme / novel dynamics**

- Whales and flash loans
- Clogged network
- Steep drop in price of ETH
- Frontrunning, backrunning, transaction ordering

[...]
3. exploitation

a.k.a. can I break this?
What we want

Reproducible attack vector

Understand impact in the system

7 phases of Smart Contract Hacking - Phase 3: exploitation
Local testing environment

Node and scripts to interact with it

Ganache (can fork mainnet!)
Geth (in dev mode)
ethers, web3.js
Buidler, Truffle
7 phases of Smart Contract Hacking - Phase 3: exploitation

Exploiting examples:

- github.com/OpenZeppelin/exploit-uniswap
- github.com/tinchoabbate/function-clashing-poc
- zpl.in/backdooring-safe-wallets
Reproduce real exploits

Execution trace:

7 phases of Smart Contract Hacking - Phase 3: exploitation

https://ethtx.info/mainnet/0xd06378b73536e7718895069a5219855774d362db47312dc304dfd4b6e39ef000
Reproduce real exploits

Execution trace:

```
[293879]: [Sender] 0x91c2d6f571d3d47a182dd59d5f41e87d4cf8e

[339502]: [Receiver] 0xe7870231992ab4baa08114faa599115fe94203f.8xfad517ac(no_ABI) => ()

[91927]: oETH $330 Put 08/14/20.addERC20CollateralOption(amtToCreate=7500000000, amtCollateral=247500000000, receiver=[Receiver] 0xe7870231992ab4baa08114faa599115fe94203f.8xfad517ac(no_ABI) => ()

[29933]: USDc.transferFrom(_from=[Receiver] 0xe7870231992ab4baa08114faa599115fe94203f, _to=oETH $330 Put 08/14/20, _value=24750.0) => (True)

[26537]: (delegate) USDc[USDcImpl].transferFrom(_from=[Receiver] 0xe7870231992ab4baa08114faa599115fe94203f, _to=oETH $330 Put 08/14/20, _value=24750.0) => (True)

[188748]: ETH 75.0 oETH $330 Put 08/14/20.execute(oTokensToExercise=1500000000, vaultsToExecuteFrom=['0xe7870231992ab4baa08114faa599115fe94203f', '0xe7870231992ab4baa08114faa599115fe94203f']) => (True)

[13085]: USDc.transfer(_to=[Receiver] 0xe7870231992ab4baa08114faa599115fe94203f, _value=24750.0) => (True)

[10216]: (delegate) USDc[USDcImpl].transfer(_to=[Receiver] 0xe7870231992ab4baa08114faa599115fe94203f, _value=247500000000) => (True)

[21485]: USDc.transfer(_to=[Receiver] 0xe7870231992ab4baa08114faa599115fe94203f, _value=24750.0) => (True)

[18616]: (delegate) USDc[USDcImpl].transfer(_to=[Receiver] 0xe7870231992ab4baa08114faa599115fe94203f, _value=247500000000) => (True)

[14497]: oETH $330 Put 08/14/20.removeUnderlying() => ()

[969]: ETH 75.0 [Receiver] 0xe7870231992ab4baa08114faa599115fe94203f.fallback() (no_ABI) => ()
```

https://ethtx.info/mainnet/0xd06378b73536e7718895069a5219855774d362db47312dc304dfd4b6e39ef000

7 phases of Smart Contract Hacking - Phase 3: exploitation

+24K USDC
4. post-exploitation

a.k.a. yeap, it’s broken. Now what?
4. post-exploitation

a.k.a. yeap, it’s broken. Now what?

wait, is this even a thing?
Don’t think of shells here :)

7 phases of Smart Contract Hacking - Phase 4: post-exploitation
Don’t think of shells here :)

**BUT**

Try to extend or elevate access

Cause further consequences in other components

Go *beyond* the exploited system
The compromised price oracle case
The compromised price oracle case

Pay ETH

Platform
selling kitties

Give away
kitties
7 phases of Smart Contract Hacking - Phase 4: **post-exploitation**

**The compromised price oracle case**

- **Buy 10 kitties**
- **Platform selling kitties**
- **Give away kitties**
- **Get price in ETH**
- **Price oracle**
The compromised price oracle case

Buy **ALL** kitties

Get price in ETH

0 ETH

Give away **ALL**
kitties for free

Platform selling kittens

Compromised price oracle

7 phases of Smart Contract Hacking - Phase 4: **post-exploitation**
The compromised price oracle case

7 phases of Smart Contract Hacking - Phase 4: post-exploitation
5 & 6 report, fixes and re-test
Public security reports!
hacking time!

github.com/tinchoabbate/defcon28-talk-challenge
Thank you!

Learn more

forum.openzeppelin.com
blog.openzeppelin.com/security-audits

Contact

@tinchoabbate