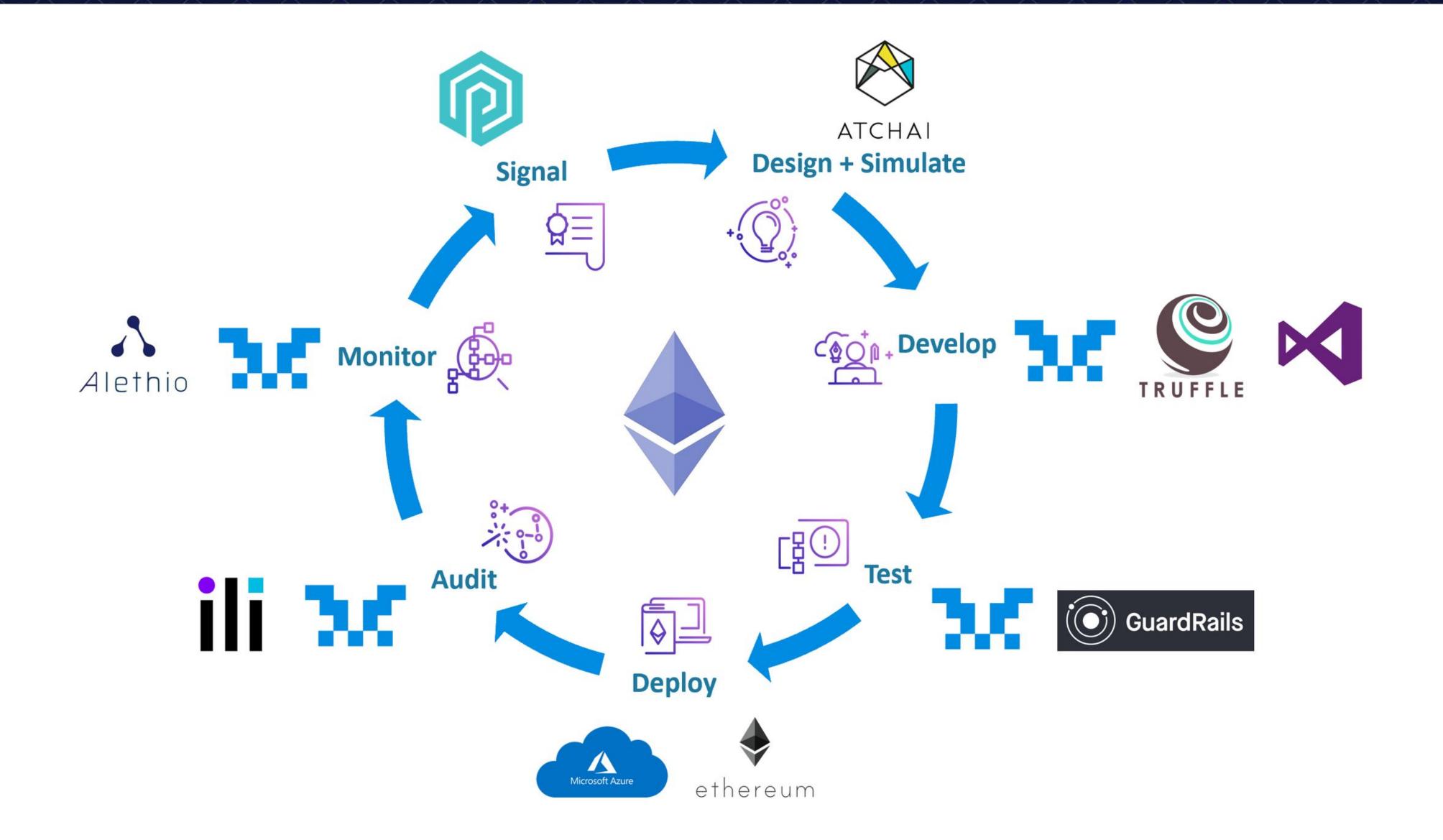
# Learn Threat Modeling Smart Contracts in <60 minutes

November 2, 2018

#### Security Activities in the SDLC



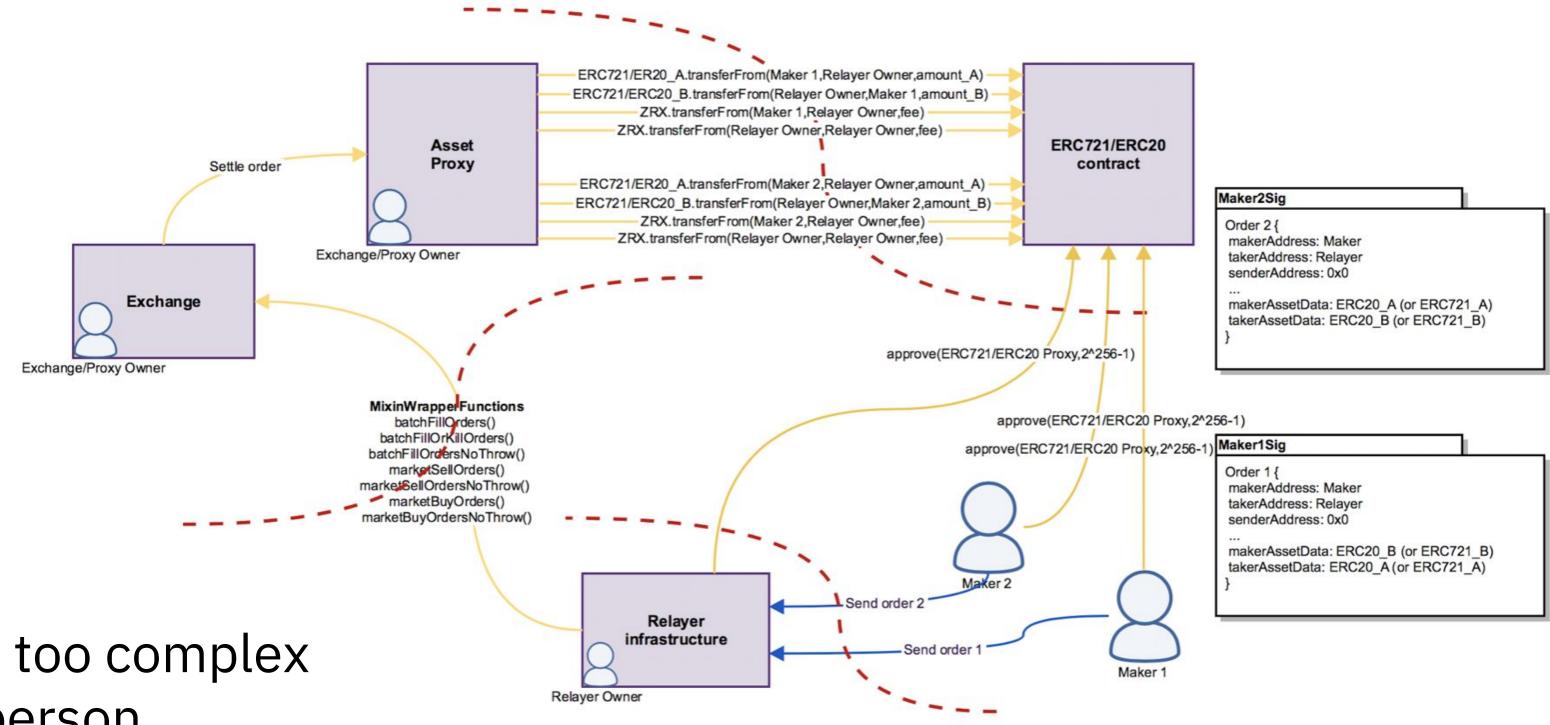
#### Formal Threat Modeling

#### Approaches

- Software-Centric
- Asset-Centric
- Attacker-Centric

#### Challenges

- Time Consuming
- For engineers too abstract / too complex
- Usually done by a security person
- Create report that nobody knows what to with



TM for 0x v2 - Relayer Matcher Strategy

#### Informal Threat Modeling

### Everybody does Threat Modeling in their heads

- Spawn an evil twin of yourself
- Get a list of threat scenarios from the evil twin
- Use the list to come up with counter strategies to something about it

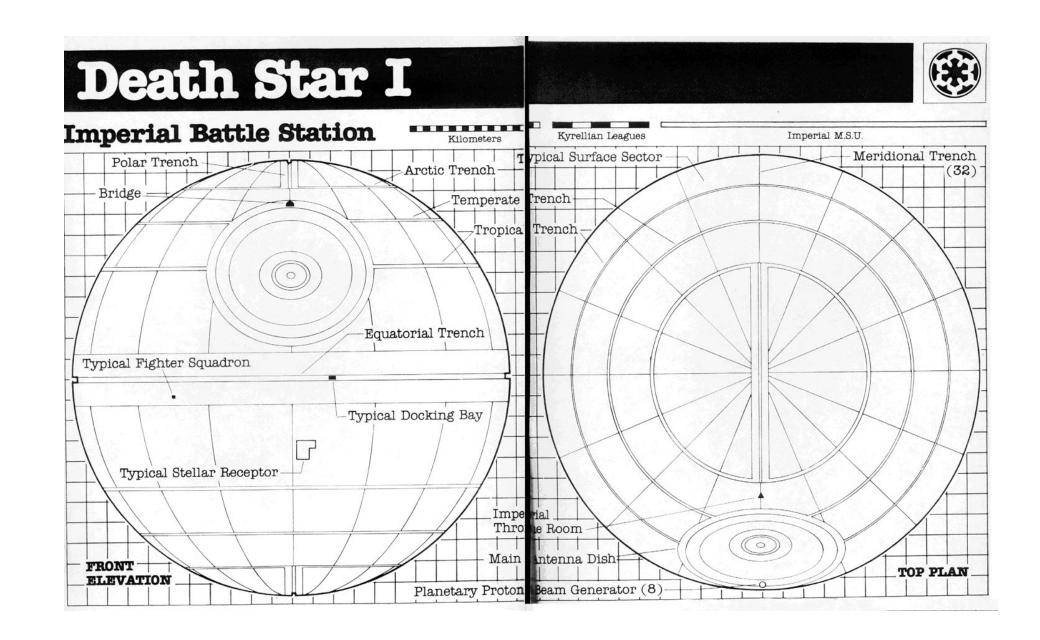
## Problem: The strategy is only as good as your evil twin

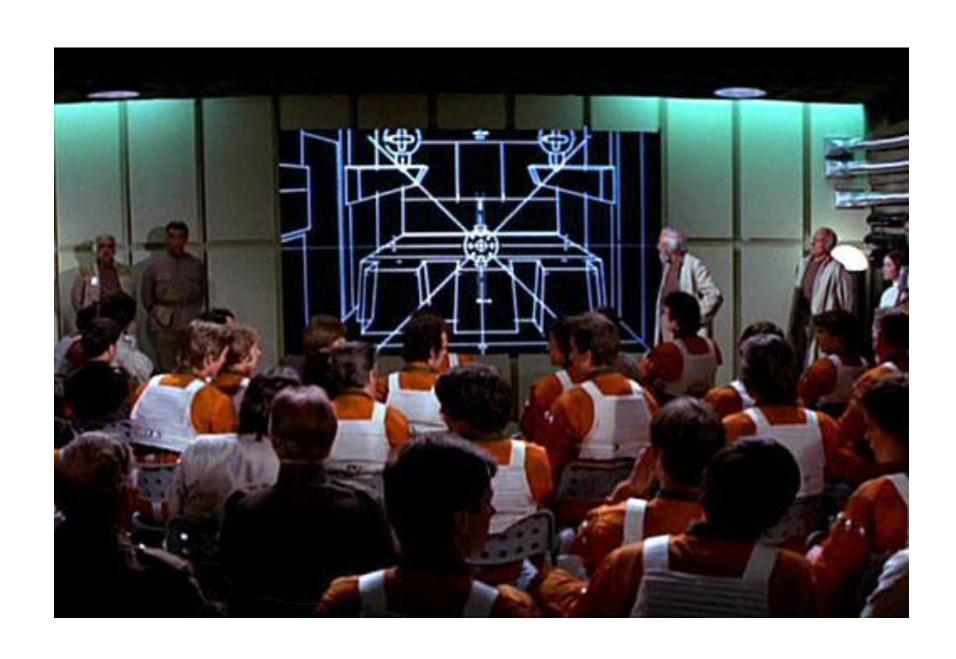


#### Threat Modeling is about

#### Answering the following 3 questions

- What are you building?
- What can go wrong?
- What are you going to do about it?





#### Finding Relevant Threats

#### TM Approach needs to:

- Do better than our evil twin
- Safe time
- Not have to learn formal TM methods

#### Use data from past blockchain security incidents

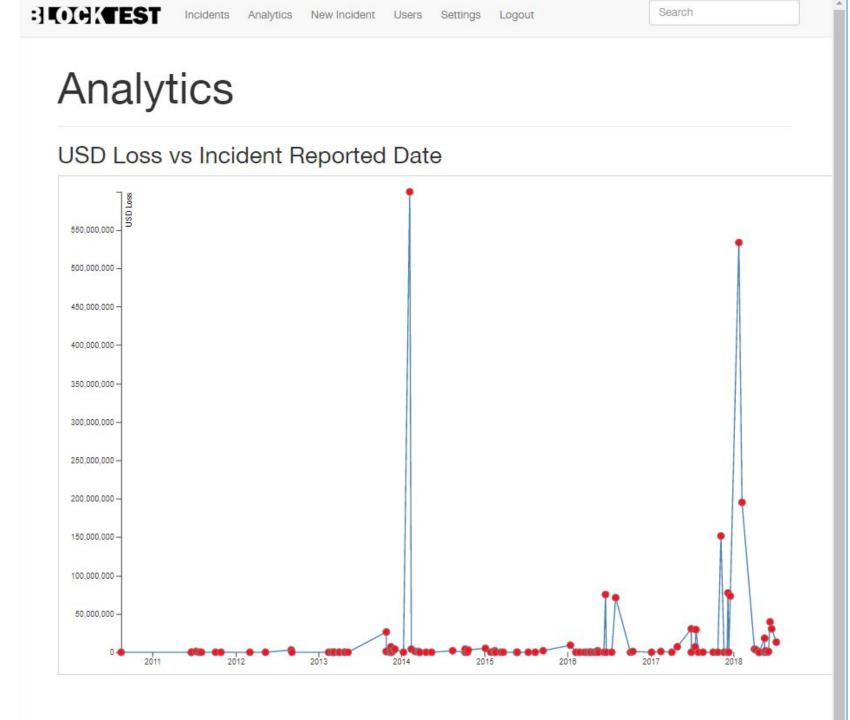
- Extract the most frequent attacks
- Categorize them based on attack patterns
- Use them to verify if they apply to your system

#### Blockchain Incident Database

- University Initiative
- 110 incidents in the database
- Nature: technical root cause, publicly reported
- Sources: Blockchain Graveyard, Reddit and other public places
- Updating incident db not straight forward
- Incident format STIX







#### Blockchain Incident Categories

#### 1. OPSEC

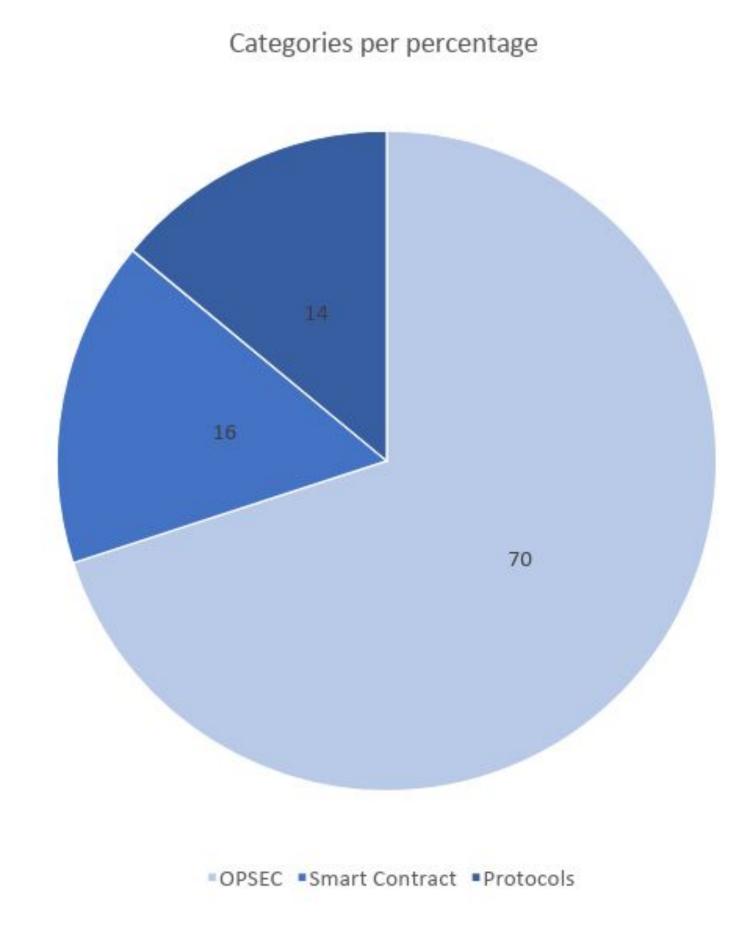
Incidents compromising an organization or individual's control of information and assets

#### 2. Smart Contract

Incidents related to bugs in smart contract systems

#### 3. Protocol

Incidents arising from malicious exploitation of protocol implementations and designs



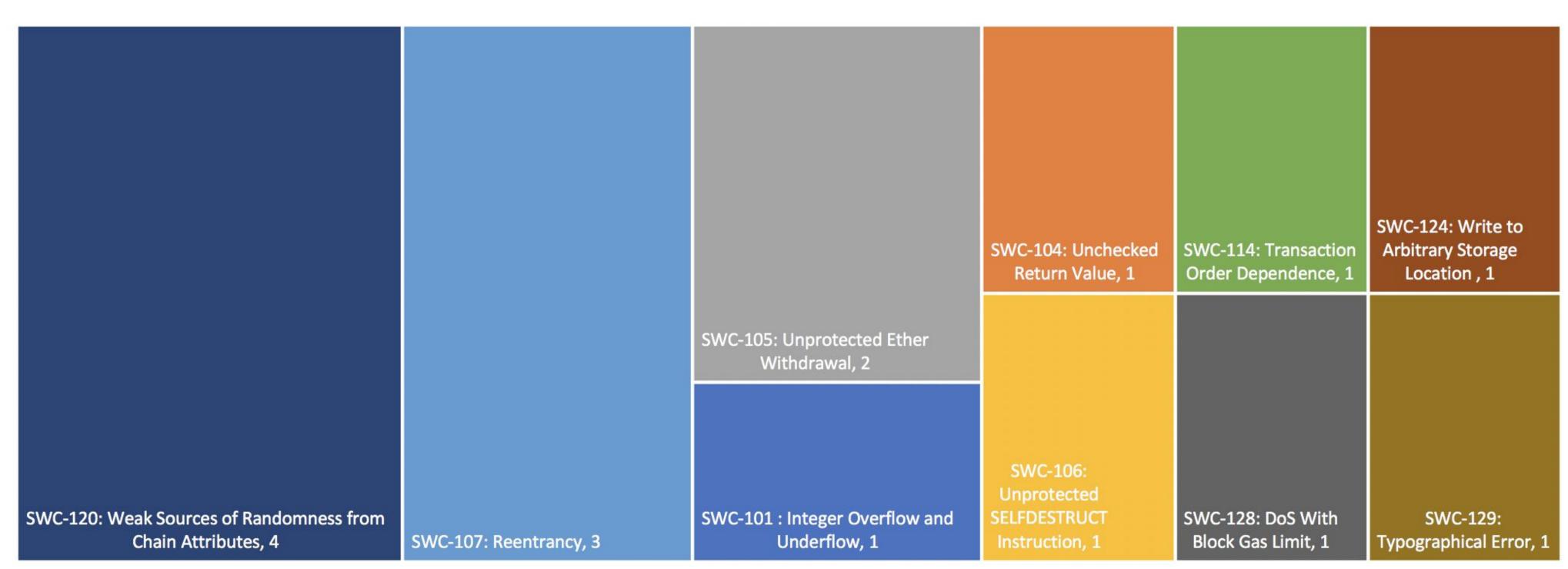
#### Smart Contract Weakness Classification (SWC)

- Scheme to classify weaknesses in smart contracts proposed in EIP-1470
- Currently in Draft
- Overlays on top of CWE
- Definitions in the SWC-registry (Github/Pages)
- Contains crafted as well as real world vulnerable contracts
- Currently 60 test samples and 30 weakness variants
- Working on Python and JS packages to access data

#### Smart Contract specific Incidents

- SWC-101 : Integer Overflow and Underflow
- SWC-105: Unprotected Ether Withdrawal
- SWC-107: Reentrancy
- SWC-120: Weak Sources of Randomness from Chain Attributes SWC-124: Write to Arbitrary Storage Location
- SWC-128: DoS With Block Gas Limit

- SWC-104: Unchecked Return Value
- SWC-106: Unprotected SELFDESTRUCT Instruction
- SWC-114: Transaction Order Dependence
- SWC-129: Typographical Error



#### Incidents categorized based on CAPEC

- CAPEC- 251: Local Code Inclusion
- CAPEC-118: Collect and Analyze Information
- CAPEC-125: Flooding
- CAPEC-148: Content Spoofing
- CAPEC-152: Inject Unexpected Items
- CAPEC-161: Infrastructure Manipulation
- CAPEC-172: Manipulate Timing and State

- CAPEC-100: Overflow Buffers
- CAPEC-122: Privilege Abuse
- CAPEC-128: Integer Attacks
- CAPEC-150: Collect Data from Common Resource Locations CAPEC-151: Identity Spoofing
- CAPEC-153: Input Data Manipulation
- CAPEC-163: Spear Phishing
- CAPEC-172: Manipulating Timing and State

- CAPEC-116: Excavation
- CAPEC-123: Buffer Manipulation
- CAPEC-129: Pointer Manipulation
- CAPEC-156: Engage in Deceptive Interactions
- CAPEC-166: Force the System to Reset Values

					CAPEC-262: Manipulate System Resources, 4  CAPEC-153: Input Data Manipulati on, 3	Input Data	CAPEC- 543: Counterfei	CAPEC- 92: Forced	172:		APEC- 6: SQL jectio	CAPEC- 194: Fak the	ke :	APEC- 128: teger
		CAPEC-156: Engage in	CAPEC-98:	CAPEC-212: Functionality Misuse, 5		Website CAPEC-	CAPEC- 123:	Leveraging		CAPEC- Force t		CAPE C	CAPE C	
CAPEC-225: Subvert Access Control 18	CAPEC-225: Subvert Access Control	Deceptive Interactions, 9	Phishing, 6	Iviisuse, 3	CAPEC-122: Privilege Abuse, 4	CAPEC-163: Spear Phishing, 3	416: Manipulat e Human Behavio	Buffer CAPEC-	CAPEC 	CAPE C	CAPE C	CAPE C	CAPE C	CAPE C
		CAPEC-255:	CAPEC-118: Collect and Analyze			CAPEC-560: Use of		116: Excavat	CAPEC 	CAPEC 	CAPE C	CAPE C	CAPE C	CAPE C
	Manipulate Data Structures, 6	Information, 5	bug, 5	CAPEC-407:	Known Domain	410: Informatio n	CAPEC- 248: Comma	CAPEC 	CAPEC 	CAPEC 	CAPE C	Doub le	Fraud , 1	
			CAPEC-21: Exploitation of	CAPEC-151:	Pretexting, 4 CAPEC-152:	CAPEC-555: Remote	Elicitatio	CAPEC-	CAPEC -22:	CAPEC 	CAPEC -94:	Sales of	frau	CAP EC-
unknown, 33	CAPEC-210: Abuse Existing Functionality, 15	CAPEC-125: Flooding, 6	Trusted	Identity Spoofing, 4	Inject Unexpect	Services with	typo, 2	100: Overflo	CAPEC 	CAPEC -17:	CAPEC -150:	Control of the Contro		12 PEC

#### Mitigation – Types of strategies

#### Acceptance: does not reduce effects

- e.g. Mitigation outweighs the cost of potential damage
- e.g. It's really unlikely that the threat occurs

## Limitation: you have one or multiple mitigation strategies to reduce risk

- e.g. Use a MultiSig wallet
- e.g. Use hardware wallets to manage private keys
- e.g. Conduct security awareness workshops

#### Transference: hand risk off to a 3rd party

e.g. Get an insurance

## Threat Modeling Exercise

#### Threat Modeling Exercise

#### Overview

- Extracted the top attack patterns from the incident db
- Find at least 3 threats and mitigation strategies
- Time: 15 minutes

#### Options

- 1. Choose your own system
- 2. Choose the sample system

#### Material

Devcon4 Github repo



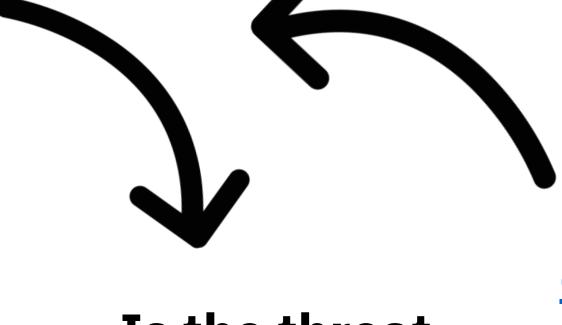
#### Threat Modeling Exercise Cheat Sheet 1/2



Take a threat extracted from the Blockchain Incident table

If yes, think about mitigation strategies and add them to your threat model

If no, move on to the next



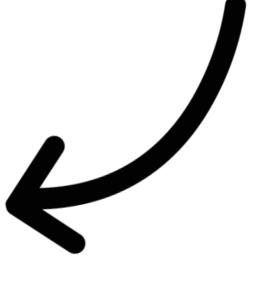
Clone the devcon4 repo

Is the threat relevant to your system?



Try to modify the threat example to make it more relevant Tip: Use your evil twin





#### Threat Modeling Exercise Cheat Sheet 2/2

#### Threat - what can go wrong?

A good threat description both describes cause and effect e.g. A hacker successfully infiltrates an employee's computer and retrieves the file containing the private keys for an account.

**Mitigation** - you know already what can go wrong so what are you doing about it? E.g. Employees use hardware wallets to make transactions and they always double check public address on the hardware device before they perform a transaction.

#### **CAPEC / Attack Pattern – how adversary**

Helps users to understand how adversaries exploit weaknesses by describing similar ways of attacking that frequently occur across a wide range of systems and technologies.

#### Threat Modeling Exercise Cheatsheet 2/2

#### Threat - what can go wrong?

A good threat description both describes cause and effect e.g. A hacker successfully infiltrates an employee's computer and retrieves the file containing the private keys for an account.

**Mitigation** - you know already what can go wrong so what are you doing about it? E.g. Employees use hardware wallets to make transactions and they always double check public address on the hardware device before they perform a transaction.

#### **CAPEC / Attack Pattern – how adversary**

Helps users to understand how adversaries exploit weaknesses by describing similar ways of attacking that frequently occur across a wide range of systems and technologies.

Go, go, go You have 15 minutes

#### Wrap Up

#### Takeaways

- OPSEC is not rocket science
- People don't pay enough attention to OPSEC
- Centralization in systems heightens the need for good OPSEC

#### Where to go from here

- Open it up to the team
- Keep your threat model up to date
- Track mitigation efforts in defect tracker

## Questions