CSE 2550: Blockchain Technology I

Lecture 13
Threat Modeling For Blockchain-based Systems

Ghada Almashaqbeh

UConn - Fall 2023

Blockchain-based Distributed Services

- Provide distributed services on top of the currency exchange medium.
  - E.g., computation outsourcing (Golem), File storage (Filecoin), video transcoding (Livepeer), etc.
- Any party can join to serve others in order to collect cryptocurrency tokens.
- The mining itself could be tied to the amount of service put in the system.
  - So servers play the role of miners.
- Several economic aspects:
  - Could provide lower cost than centralized service providers.
  - A step forward on the “useful mining” path.
  - Utility tokens vs. store of value tokens.

But … Are They Secure?!

- The blockchain space experienced a huge number of attacks.
    - Financial incentives lead to more motivated attackers.
- Security is even more challenging in blockchain-based distributed services.
    - Complicated functionality.
    - Larger scale.
    - Usually open access model, anyone can join with no pre-identification/authentication.
    - Fair service-payment exchange is impossible between distrusted parties.
    - Performance issues may lead to sacrificing security for efficiency.

Threat Modeling

- Threat modeling is an essential step in secure systems design.
  - Explore the threat space and identify all potential attack scenarios.
  - Helps in both guiding the system design, and evaluating the security level of developed systems.
- Not only secure systems design ...
  - Essential for theory and applied cryptography, secure software design, communication protocols, storage systems, hardware design, and the list can go on and on.
  - Primitives, protocols, hardware components, etc., do not work in a stand alone model.
    - In practice, we have parallel and sequential composability.

Are Blockchain Systems Any Different?

- Existing solutions: a popular example is STRIDE from Microsoft
  - Stands for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege.
  - Mainly for secure software development.
- Many other tools/frameworks exist, but all are also limited.
- Traditional approaches do not fit blockchain-based systems.
  - Do not scale.
  - Do not explicitly account for attacker financial motivations.
  - Do not explicitly account for collusion between attackers.
  - Do not consider the new threat types that cryptocurrencies and blockchains introduce.

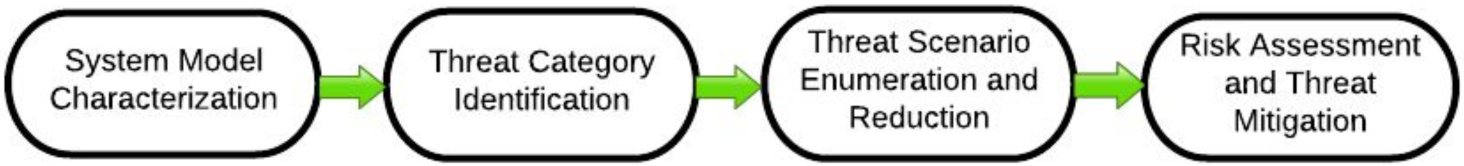
ABC: Asset-Based Cryptocurrency-focused Threat Modeling Framework*

*Ghada Almashaqbeh, Allison Bishop, and Justin Cappos. "ABC: A Cryptocurrency-Focused Threat Modeling Framework." in INFOCOM-CryBlock (2019).

What is ABC?

- A systematic threat modeling framework geared toward blockchain-based systems.
  - Its tools are useful for any distributed system.
- Helps designers to focus on:
  - Financial motivation of attackers.
  - New asset types in cryptocurrencies/blockchains.
  - Deriving system-specific threat categories.
  - Spotting collusion and managing the complexity of the threat space.
    - Done using a tool called a collusion matrix.
- Integrates with other steps of a system design; risk management and threat mitigation.

ABC Steps



Running Example: CompuCoin

- A cryptocurrency that provides a distributed computation outsourcing service.
- Parties with excessive CPU power may join as servers to perform computations for others in exchange for CompuCoin tokens.
- The mining process is tied to the amount of service these servers provide.

Step 1: System Model Characterization

- Identify the following:
  - Activities in the system.
    - Or system modules.
  - Participant roles.
  - Assets: important components that if compromised the security of the system will be compromised.
  - Any external dependencies on other services/systems/parties.
  - System assumptions.
- Draw a network diagram(s) of the system modules.

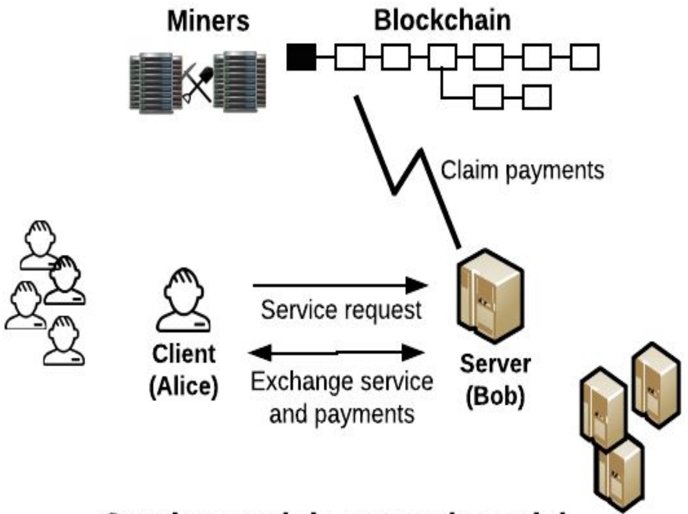
Step 1: Running Example Application

Functionality description. Outlined in CompuCoin description introduced earlier.

Participants. Clients and servers.

Dependencies. May rely on a verifiable computation outsourcing protocol.

Assets. Computation service, service rewards (or payments), blockchain, currency, transactions, and the communication network.



Service module network model

Step 2: Threat Category Identification

- Define broad threat classes that must be investigated.
- ABC defines these classes in an asset-focused way.
- For each asset, do the following:
  - Define what constitutes a secure behaviour for the asset.
    - For some assets, there could be a well-defined security notion in the literature.
    - Others, you cna to come up with (perhaps intuitive) security notion.
  - Use that knowledge to derive the asset security requirements.
  - Define threat classes as violations of these requirements.
    - I.e., inverting a requirement produces a threat category.

Step 2: Running Example Application I

- Apply step 2 to each of the assets in CompuCoin:
  - Service (computation outsourcing).
  - Service rewards.
  - Blockchain.
  - Currency.
  - Transactions.
  - Communication network.
- Step 2 produces the threat category table found in the next slide.

Step 2: Running Example Application

Asset	Security Threat Category
Service	Service corruption (provide corrupted service for clients).
	Denial of service (make the service unavailable to legitimate users).
	Information disclosure (service content/related data are public).
	Repudiation (the server can deny a service it delivered).
Service payments	Service slacking (a server collects payments without performing all the promised work).
	Service theft (a client obtains correct service for a lower payment than the agreed upon amount).
Blockchain	Inconsistency (honest miners hold copies of the blockchain that may differ beyond the unconfirmed blocks).
	Invalid blocks adoption (the blockchain contains invalid blocks that do not follow the system specifications).
	Biased mining (a miner pretends to expend the needed resources for mining to be elected to extend the blockchain).
Transactions	Repudiation (an attacker denies issuing transactions).
	Tampering (an attacker manipulates the transactions in the system).
	Deanonymization (an attacker exploits transaction linkability and violates users' anonymity).
Currency	Currency theft (an attacker steals currency from others in the system).
Communication network	Denial of service (interrupt the operation of the underlying network).

Lack of progress (or DoS).

Information disclosure.

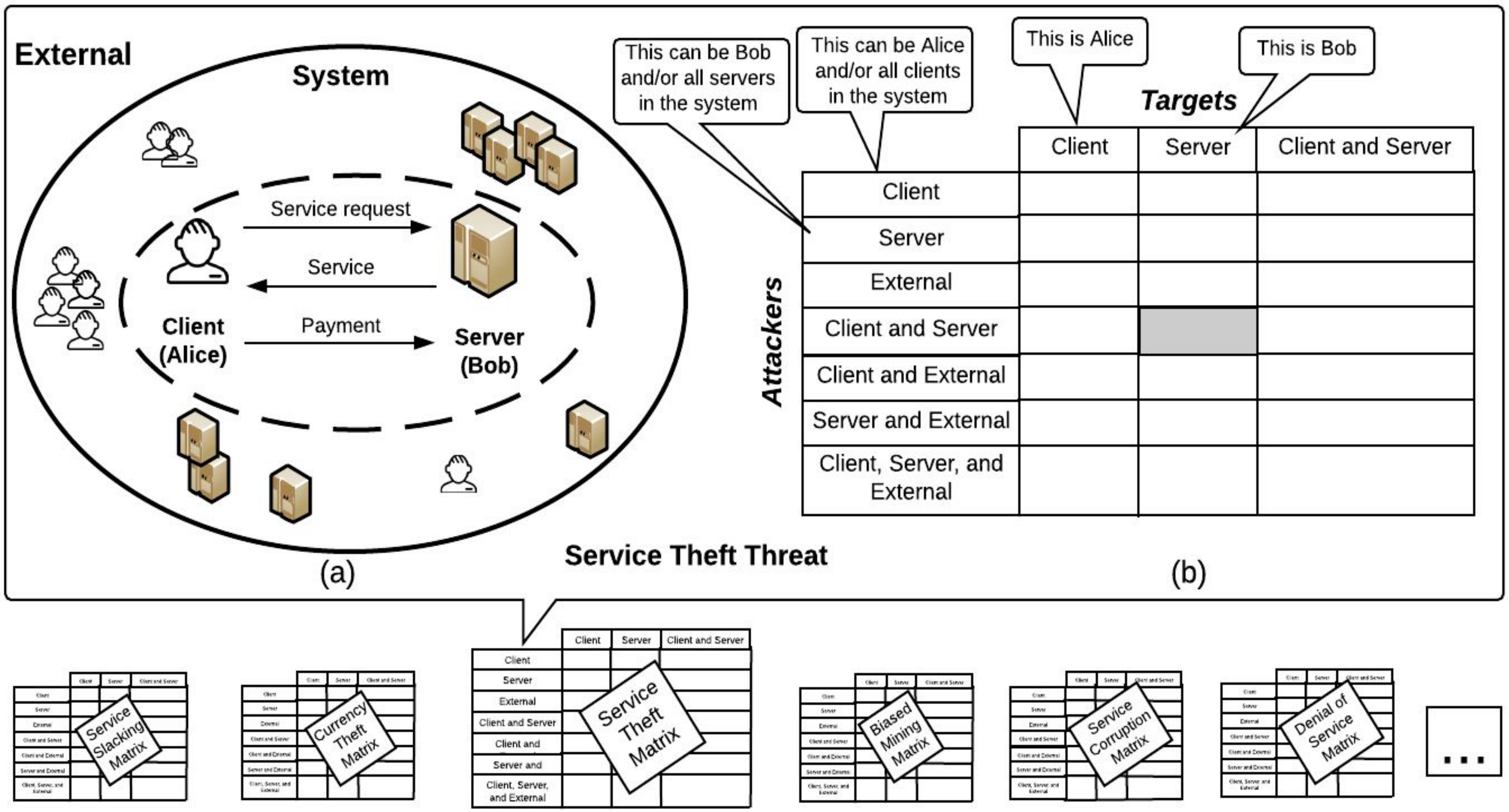
Step 2 - More

- Although it was produced for CompuCoin, this table is quite generic and can be applied to other systems as well.
- More categories could be added or removed depending on the system under design and the amount of information you have about the system.
  - Does Bitcoin need all the categories listed in the previous table?
  - What if a system provides more than one type of service? (e.g., Filecoin provides a file storage and retrieval services.)

Step 3: Threat Scenario Enumeration and Reduction

- For each threat, define scenarios that attackers may follow to pursue their goals.
  - Be comprehensive as possible, consider collusion and financial motivations.
- ABC devises collusion matrices to help with this step.
- Analyzing a collusion matrix involves:
  - Enumerating all possible attack scenarios.
  - Crossing out irrelevant cases and merge together those that have the same effect.
  - Documenting all distilled threat scenarios and the reasons behind deletion/merging.
    - This is the outcome of the threat modeling process.

Collusion Matrix



Step 3: Running Example Application

Service Theft Threat Collusion Matrix

<table>
<tr><th>Target
Attacker</th><th>Client</th><th>Server</th><th>Client and Server</th></tr>
<tr><td>External</td><td rowspan="7">Clients cannot be targets because they do not serve others.</td><td rowspan="3">Servers and external cannot attack because they do not ask/pay for service.</td><td rowspan="7">Reduced to the case of attacking servers only, clients do not serve others (cannot be targets).</td></tr>
<tr><td>Server</td></tr>
<tr><td>Server and External</td></tr>
<tr><td>Client</td><td>(1) Refuse to pay after receiving the service.
(2) Issue invalid payments.</td></tr>
<tr><td>Client and External</td><td rowspan="3">Reduced to the case of an attacker client. A client does not become stronger when colluding with other servers or external entities.</td></tr>
<tr><td>Server and Client</td></tr>
<tr><td>Client, Server, and External</td></tr>
</table>

Step 4: Risk Management and Threat Mitigation I

- An independent task of threat modeling.
- Important questions to answer:
  - Do I need to address threats?
  - Is there a priority order for addressing these threats?
    - Based on threat impact, for example.
  - Any Security-efficiency trade-offs?

Step 4: Risk Management and Threat Mitigation II

- Financial incentives affect prioritizing threats and their mitigation techniques.
  - Use game theory-based analysis to quantify the utility/profits an attacker may obtain.
  - Use detect-and-punish techniques to address certain threat types.
  - Devise algorithms/proofs/etc. that are more profitable (in terms of resources) when executed in an honest way than when executed in a malicious way.
- For example, in CompuCoin:
  - Locking payments in an escrow neutralizes threat 1.
  - Having a penalty deposit that is fortified upon cheating addresses threat 2.
  - Both require careful design and economic analysis.

An Iterative Process

- Any alteration on the system design requires revisiting the threat modeling step.
  - Efficiency optimizations, building block replacement, introducing extra dependencies in the system, etc.
- Assess the system security level in the after design stage.
- Care must be taken with respect to financial threats.
  - Attacker's incentives may change over time, which may impact the economic threat mitigation techniques or even change the risk level of a threat.
  - Accounting for the fact that several external, perhaps competing, systems exist.