Compliance Checking for DApp Analysis
A Case Study on the Beanstalk Attack

F. Corradini, A. Galassi, A. Marcelletti, L. Migliorelli, A. Morichetta and B. Re

{flavio.corradini, alessio.galassi, alessand.marcelletti, andrea.morichetta, barbara.re}@unicam.it

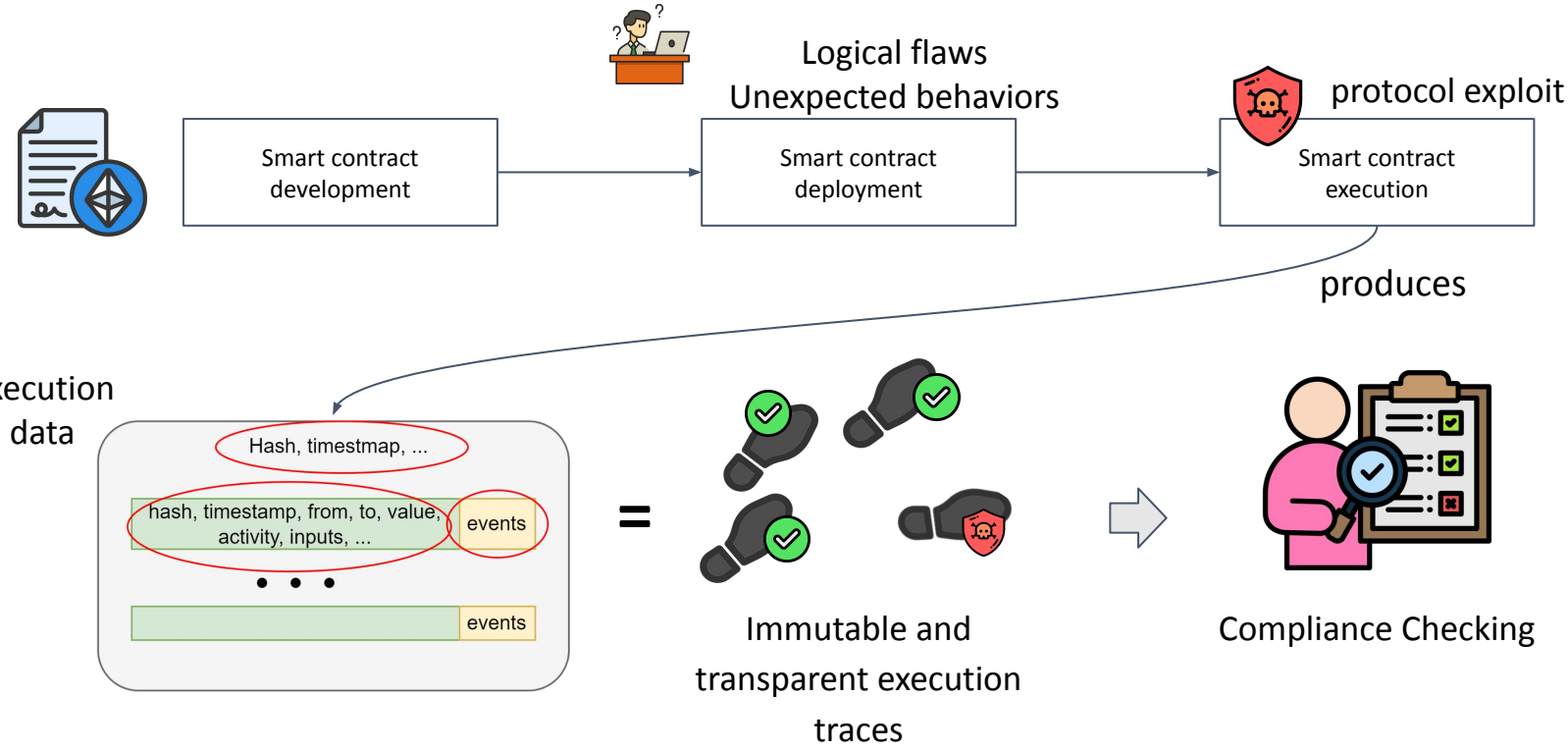
Computer Science Division, School of Science and Technology,

University of Camerino, Italy

8th Distributed Ledger Technology Workshop

Pula, June 5th 2026

Context



RQ: How can compliance checking be adopted to analyze a governance attack?

The Beanstalk Case Study

Beanstalk is a credit-based **stablecoin protocol on Ethereum** with a **governance system** where users can make proposals and vote on them

It was the victim of a **$181M hack** on April 17, 2022

The attacker executed a **flash loan governance attack**, temporarily borrowing over $1 billion worth of tokens within a single transaction

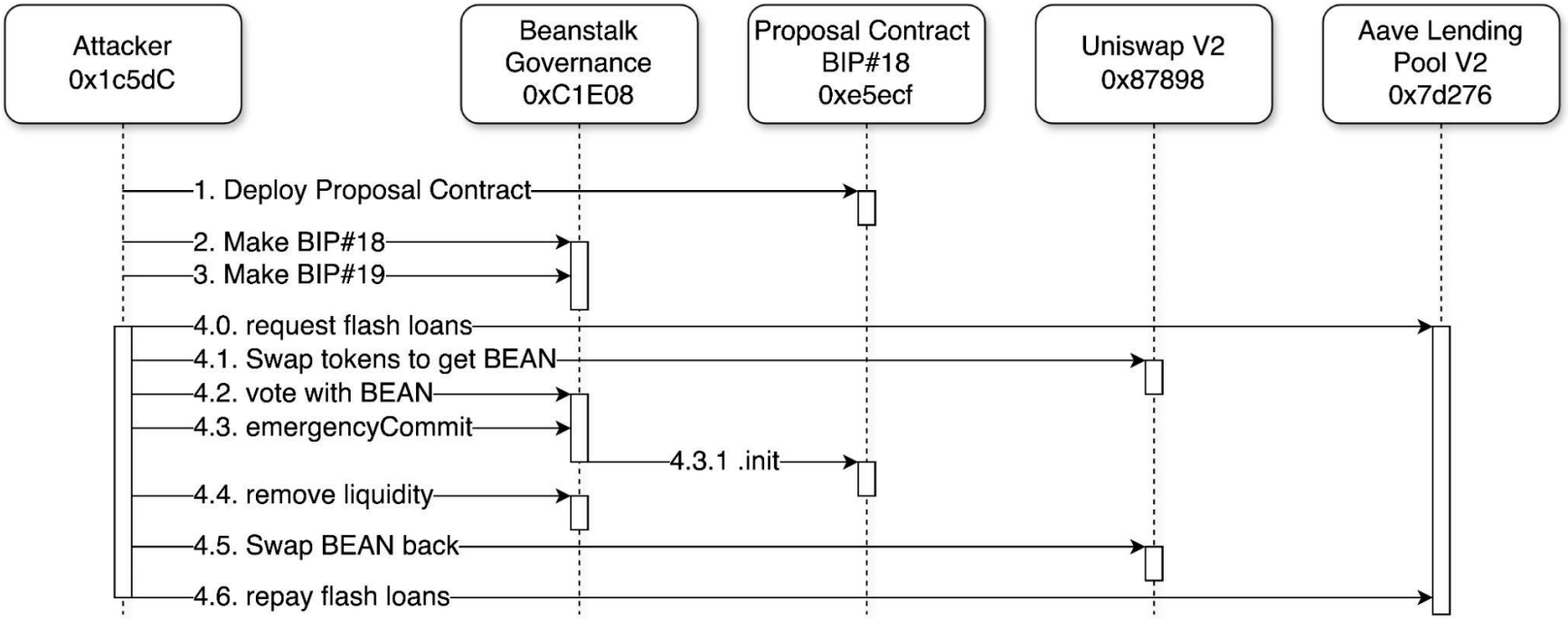
DAO, Governance & Flash Loans

A **Decentralized Autonomous Organization (DAO)** is an organizazion governed by rules encoded in smart contracts rather than by centralized authorities

Governance protocols mechanisms that enable on-chain decision-making, allowing token holders to vote on proposals and influence protocol actions.

Flash loan is a DeFi-native lending mechanism that allows users to borrow uncollateralized funds of arbitrary size, the loan is repaid within the same transaction.

Anatomy of an Exploit



Compliance Checking & CoBlock

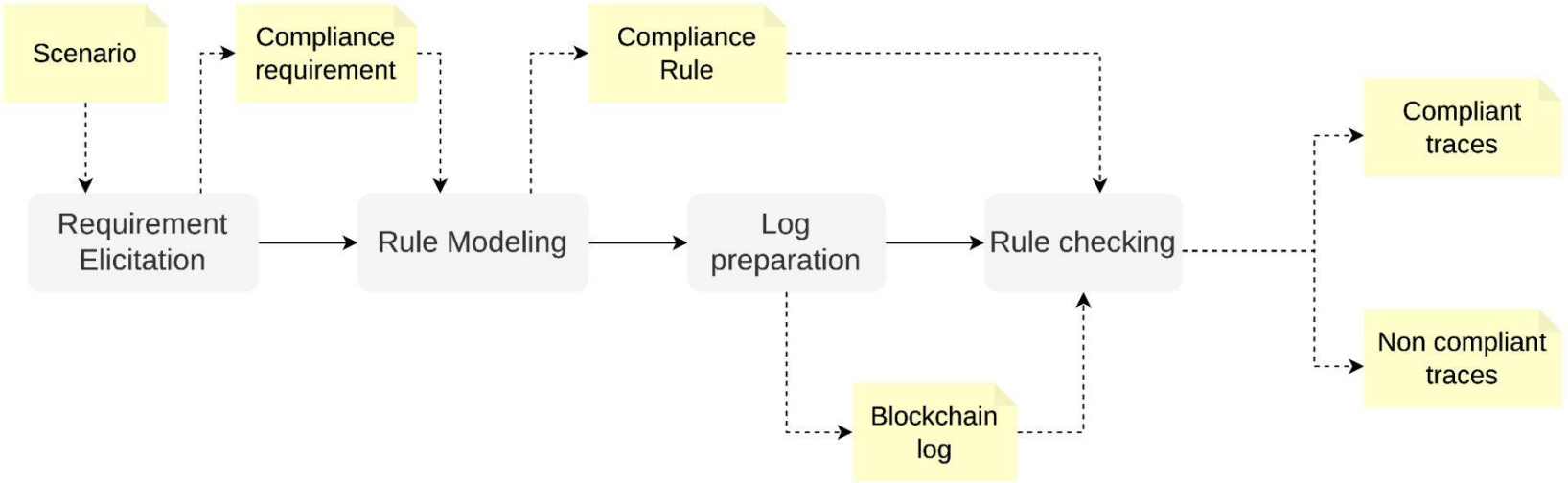
Compliance checking is a process mining technique, it aims at verifying the adherence of rules over business process models or execution data

CoBlock is a domain-specific language for modeling compliance rules in terms of blockchain characterizations, treated as first-class citizens

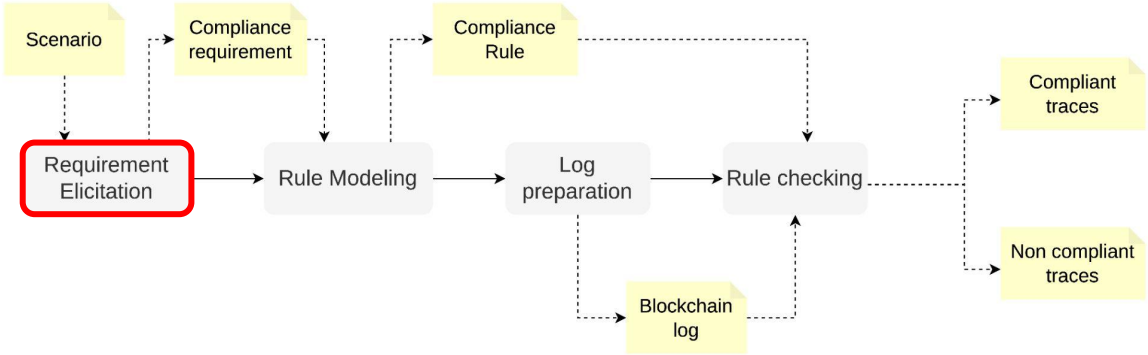
$$
\begin{aligned}
\langle \text{RULE} \rangle &::= \langle \text{TX} \rangle \langle \text{CF}_u \rangle \\
\langle \text{CF}_u \rangle &::= \texttt{occ} \mid \texttt{nocc} \\
\langle \text{TX} \rangle &::= \textit{tx}(\langle \text{ATTR} \rangle) \\
\langle \text{OP} \rangle &::= \texttt{is} \mid \texttt{is not} \\
\langle \text{ATTR} \rangle &::= \langle \text{A} \rangle \mid \langle \text{A} \rangle \langle \text{ATTR} \rangle \\
\langle \text{A} \rangle &::= \langle \text{CALL} \rangle \mid \langle \text{E} \rangle \mid \lambda \\
\langle \text{CALL} \rangle &::= \langle \text{OP} \rangle \texttt{ called } \textit{call}() \\
\langle \text{E} \rangle &::= \langle \text{OP} \rangle \texttt{ emitted } \textit{event}()
\end{aligned}
$$

F. Corradini, A. Galassi, A. Marcelletti, B. Re, Coblock: a domain-specific language for compliance checking on smart contract execution data, BCRA (2025).

Methodology



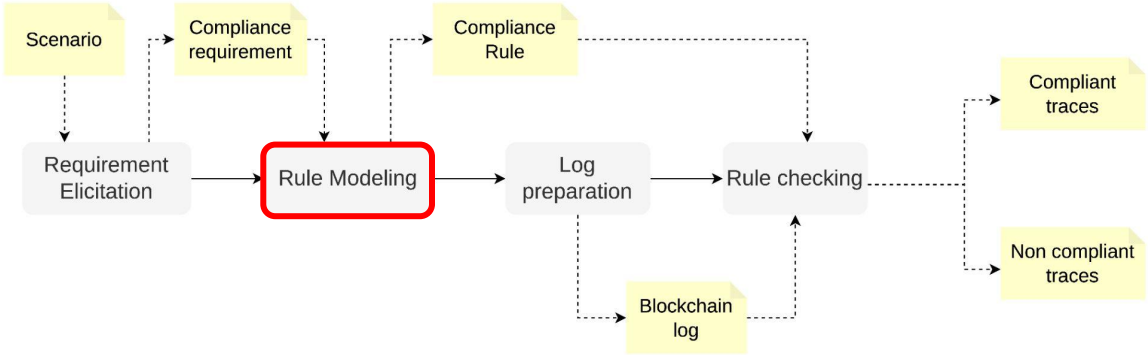
Methodology: Requirement Elicitation



To define the **points of interest that should be monitored in an application**, such as protocol guidelines or anomaly thresholds, independently of specific attack paths.

Governance improvement proposal votes should not be manipulated

Methodology: Rule Modeling

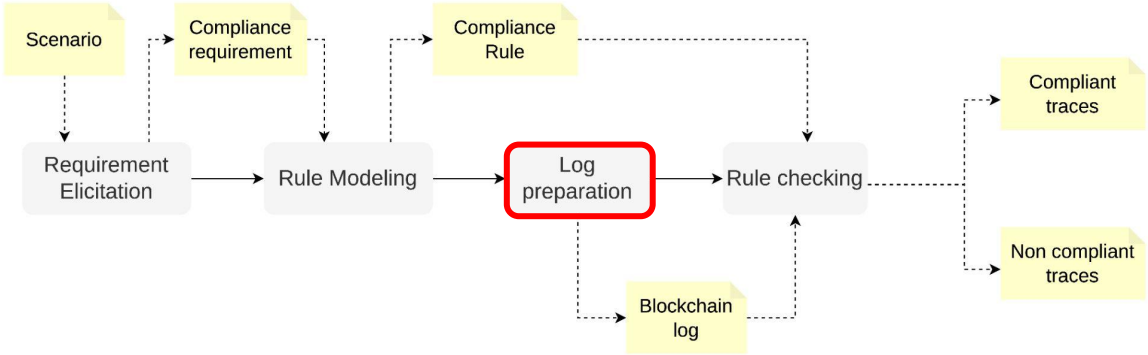


Represent compliance requirements as formal specification that can be automatically verified

```
voteTX(
    is called vote()
    is emitted FlashLoan()
) occ
```

F. Corradini, A. Galassi, A. Marcelletti, B. Re, Coblock: a domain-specific language for compliance checking on smart contract execution data, BCRA (2025).

Methodology: Log Preparation

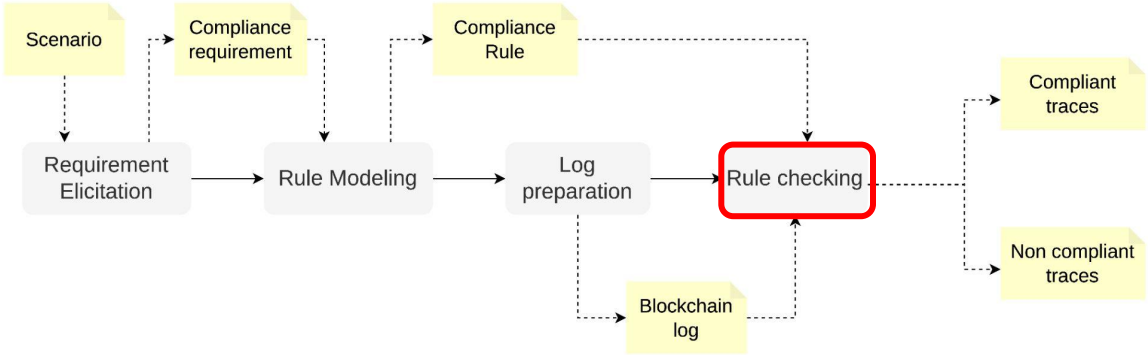


Smart contract execution data is retrieved and processed to obtain XES log used to verify the rules

1. Data extraction
2. **Case identifier** selection
3. Execution trace reconstruction

F. Corradini, A. Marcelletti, A. Morichetta, B. Re, A data extraction methodology for ethereum smart contracts, in: International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events, IEEE, 2024, pp. 524–529.

Methodology: Rule Checking



The rule-checker engine iterates over every trace in the blockchain log and, for each trace, it checks whether the matching conditions specified in the rule are satisfied

Subset of traces matching rule's condition

Subset of traces <u>not matching</u> rule's condition

Compliance Checking for DApps in action

CoBlock framework

Blockchain log

Choose file:

Choose File | beanstalk-full-sender.xes

Upload log

beanstalk-full-sender.xes

N° events loaded

2421

N° traces loaded

841

Blockchain log mapping

CA: contractAddress

B: blockNumber

S: sender

Blockchain-based compliance rules

Rule:

voteTX(is called vote() is emitted FlashLoan()) occ

Rule parser

Parse rule

Rule "voteTX(is called vote() is emitted FlashLoan()) occ" successfully compiled!

Check rule over blockchain log

[preview] Compliant traces - Download full JSON

object ▶ 0 ▶ 0 ▶

▼ object {1}

▼ 0 [3]

▼ 0 {15}

functionName : propose

(the attack)

[preview] Non-compliant traces - Download full JSON

Select a node...

▼ object {20}

▶ 0 [3]

▶ 1 [1]

▶ 2 [1]

(other legit executions)

Rule Modeling Decision

```
voteTX(
     is called vote()
     is emitted FlashLoan()
) occ
```

Precision
It would only fit the specific case study

Generalization
It is applicable to a wide range of DeFi governance systems

Log Preparation

Case concept selection directly impact the trace granularity and the analysis perspective

Tx hash
One trace per transaction, focus on single tx in isolation

Sender addr.
Group txs from the same sender, analyze operation by account

Block no.
Group txs in the same block, analysis on inclusion criteria

Contract addr.
Group txs on the same SC, analyze interaction patterns on it

Conclusions

We explored the adoption of compliance checking as a technique for analyzing governance attacks in decentralized applications.

Future Works:

Attack pattern collection
Build an extended collection of compliance rule pattern modeling know attack

Active monitoring
Shift the rule checking as soon transaction are sent to the mempool

Thank you!

Andrea Morichetta

andrea.morichetta@unicam.it