Transparent Protection of Aggregate Computations from Byzantine Behaviours via Blockchain

Danilo Pianini*, **Giovanni Ciatto***, Roberto Casadei*,

Stefano Mariani§, Mirko Viroli*, Andrea Omicini*

* Dipartimento di Informatica, Scienza e Ingegneria (DISI)
Alma Mater Studiorum — Università di Bologna

§ Dipartimento di Scienze e Metodi dell'Ingegneria (DISMI)
Università di Modena e Reggio Emilia

Aggregate Computing (AC)

What is it?

Aggregate Computing (AC) is a **computational model** which

- directly targets *aggregates* (i.e., **ensembles of** computational **devices**)
- the system consists of several **interconnected & communicating** devices, **situated** into the *physical* space
  - devices can **perceive** their *environment* and keep up with their **neighbours**
- humans **program** the aggregate $\longrightarrow$ a **compiler** instructs each device
- the aggregate's **global** behaviour is an **emergent** phenomenon
- produced by the devices **periodically** executing the same program
  - in a **controlled & reproducible** way
- usual *computational* constructs + a fixed set of **local**-only operators

Aggregate Computing (AC)

Why is it so cool?

In general

- AC makes software applications **very robust** & **massively scalable**
- Perfect deal for large- up to **wide-scale**, **space-aware** applications
  - implying some **spatially** situated *sensing* or *acting*
- Expectations for what concerns **safety-critical** scenarios

Aggregate Computing (AC)

Why is it so cool?

In general

- AC makes software applications **very robust** & **massively scalable**
- Perfect deal for large- up to **wide-scale**, **space-aware** applications
  - implying some **spatially** situated *sensing* or *acting*
- Expectations for what concerns **safety-critical** scenarios

Examples

- People **wayfinding** or **tracking** within smart cities
- **Crowd** management during public events, or dangerous situations
- Autonomous- or smart-vehicles **steering**
- **Rescue operations** within open spaces

Aggregate Computing (AC)

Example: Crowd detection

Aggregate Computing (AC)

Example: Crowd detection

*Sounds cool. But it's quite **insecure**. Let's see why!*

Aggregate Computing (AC)

Local VS Global

- *LEFT*: this is how the system actually **is**
- *RIGHT*: developers can reason in terms of **points**, **space**, and **time**
  - Devices can only **communicate** with their **neighbours**

Aggregate Computing (AC)

About devices inter-communication

Aggregate **algorithms** require devices to communicate with **neighbours**:

Aggregate Computing (AC)

About devices inter-communication

Aggregate **algorithms** require devices to communicate with **neighbours**:

- the AC **compiler** makes all devices execute the same **program**
- the same program is **periodically** re-executed, taking as input
  - the local **sensor** data
  - the **output value** of the previous round, if any
- each device **sends** its last output value to its **current neighbours**
  - at the end of each round
- often, neighbours values are **aggregated** on a neighbourhood-wise basis
  - e.g. `min`, `max`, `avg`, `count`, and other aggregation operators
- these are the basic **bricks** for building space-related computations

Aggregate Computing (AC)

About devices inter-communication (Graphically)

Aggregate Computing (AC)

About devices inter-communication (Graphically)

Aggregate Computing (AC)

About devices inter-communication (Graphically)

The problem with AC: Byzantine Behaviours

What prevents devices from deviating from the protocol?

The problem with AC: Byzantine Behaviours

What prevents devices from deviating from the protocol?

Nothing.

The problem with AC: Byzantine Behaviours

What prevents devices from deviating from the protocol?

Nothing.

Editing the AC program? Not always works…

The problem with AC: Byzantine Behaviours

What prevents devices from deviating from the protocol?

Nothing.

Selective (or Byzantine) Attacks

- Where malicious devices send **different** messages to **each** neighbour
  - rather than performing *multi-casts* as dictated by AC

The problem with AC: Byzantine Behaviours

What prevents devices from deviating from the protocol?

Nothing.

Masquerade Attacks

- Where malicious devices fake their identity **impersonating** other devices
  - e.g. by stealing their identifiers

The problem with AC: Byzantine Behaviours

What prevents devices from deviating from the protocol?

Nothing.

Sybil Attacks

- Where malicious devices fake **multiple identities** in order to get **disproportionate weight** in consensus- or voting-based algorithms

The Blockchain

- The Blockchain is an **emerging & disruptive** means for **distributing computation** among mutually-**untrusted** parties
- It usually consists of a **State Machine Replication** system aimed at maintaining a *shared*, *append-only* **ledger** for **transactions**
  - transaction = an **event**, possibly changing the state of the system
- Several clones of the same program (**replicas**) are in charge of **notarising** the many transactions affecting some distributed system's **global** state
  - notarise = agree on **who** did **what** and **when**
- **Untamperable** transactions history: **replicated** and cryptographically secured ledger, through **hash-functions**
  - **signed** transactions provide **accountability & non-repudiation**

The Blockchain

Our proposal

Let's use the Blockchain as the underlying communication mean for AC devices

The Blockchain

Our proposal

> *Let's use the Blockchain as the underlying communication mean for AC devices*

The Blockchain as the communication layer for AC

- AC devices may have a particular **identity** in the eyes of the Blockchain system
- **Masquerade & Sybil** attacks are cut off by construction
- Replicas may be easily programmed to filter out **Selective** attacks
- The Blockchain layer may enforce additional, complex **validation policies**
  - e.g. filtering out nodes lying about their phisical position
- In any case: **the problem is solved trasparently w.r.t. AC**

The proposed architecture

Layered P.o.V.

The proposed architecture

The network P.o.V.

The proposed architecture

Problem: Which Blockchain technology?

*Permission**ed** VS Permission**less***

≈

***With** cryptocurrency VS **Without** cryptocurrency*

The proposed architecture

Problem: Which Blockchain technology?

*Permission**ed** VS Permission**less***

≈

***With** cryptocurrency VS **Without** cryptocurrency*

Implications

- *Permission**less***:
  - cryptocurrency at app level + low throughput + openness
- *Permission**ed***:
  - no money involved + high throughput + access control

Conclusions

- AC is a great tool for **mission critical** apps involving several **devices**
  - and, of course, their owners, i.e. **people**
- Currently, a malicious attaker may subvert an aggegate system through **Byzantine behaviours**
  - which may be harmful
- We identifyied the set of Byzantine behaviours possibly affecting AC
  - i.e., the **Selective**, **Masquerade**, and **Sybil** attacks
- We propose a **Blockchain-based** architecture aimed at solving such issues

Future Works

- Simulating a Blockchain system over the Alchemist simulator
- Simulating the proposed Blockchain-based architecture & Byzantine behaviours
- Providing a **Proof-of-Concept** implementation over a real-world Blockchain technology
    - MultiChain, and HyperLedger Fabric are our candidates

Thank you for your attention!