Edge-Cloud hybrid Model for Distributed Applications

Albert van der Linde
a.linde@campus.fct.unl.pt
NOVA LINCS - DI, FCT, Universidade NOVA de Lisboa

Distributed Systems
(peer-to-peer; data-consistency; security)

Advisors:
Nuno Preguiça & João Leitão

Introduction

1) What is the problem that you are actually going to solve in your work?

Interactive applications are typically built via the client-server communication model.

This model limits interactivity as all interactions have to go through the client-server-client route.

2) Why is it a problem [why does anybody else care]?

Large amount of interacting clients:

Hard to scale, even using cloud services (and expensive!).

No support for disconnection from a server.

High latency between users, especially noticed if users are close-by.

Introduction

3) A 1-sentence positive, startling statement about your work that will address this problem.

Move away from the client-server model towards a cloud-edge hybrid.

Bring application logic and data to the client side and propagate operations directly between clients (peer-to-peer fashion).

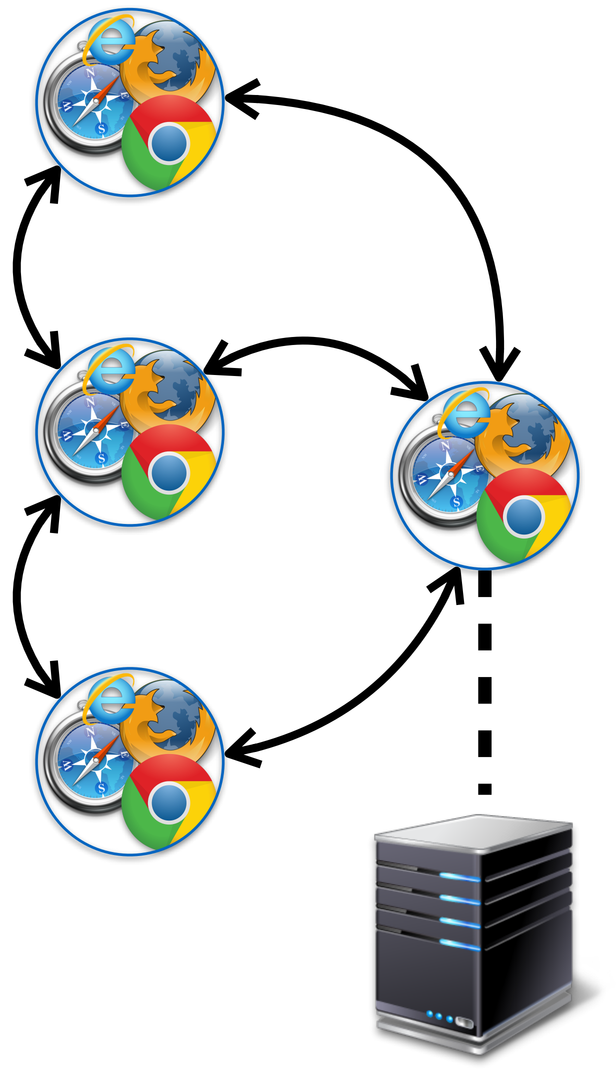
4) What's the consequence of the startling statement [in addressing the problem]?

Having clients interacting directly brings two main challenges:

dealing with many writes nodes (concurrency, data-consistency);

dealing with misbehaving users (application security).

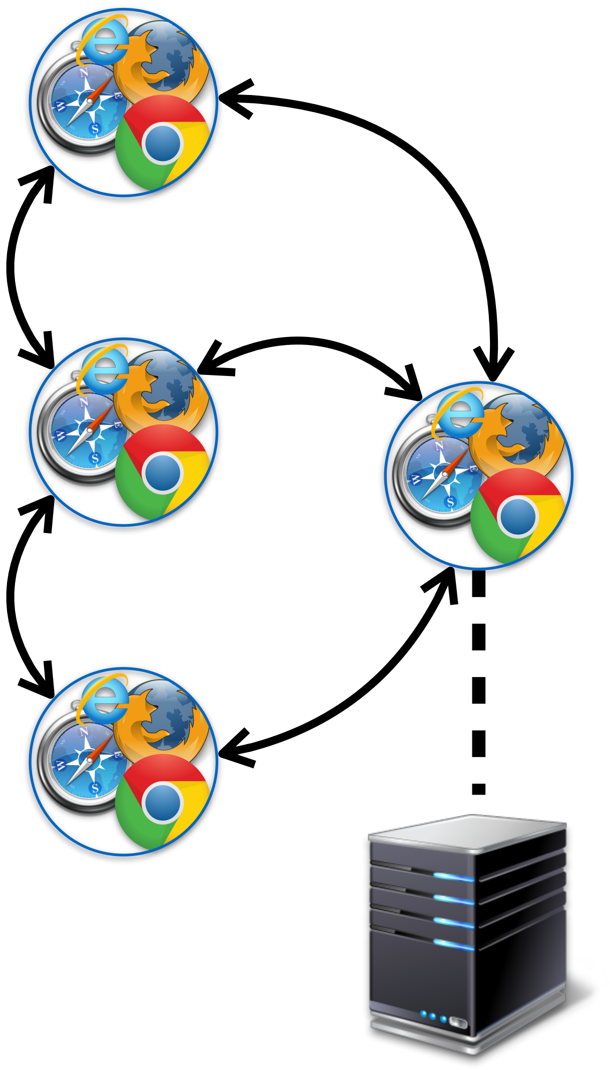
Legion



- What we have
  - Legion [1]- Framework to develop interactive web-applications
    - Shared data-structures (lists, maps…)
    - Peer-to-peer connections over WebRTC

[1] https://legion.di.fct.unl.pt
Legion: Enriching internet services with peer-to-peer interactions. WWW '17. Albert van der Linde et al.

Legion



- What we have
  - Legion [1]- Framework to develop interactive web-applications
    - Shared data-structures (lists, maps…)
    - Peer-to-peer connections over WebRTC
  - Improved scalability
  - Lower dependency on the server
  - Lower latency (user-user)

[1] https://legion.di.fct.unl.pt
Legion: Enriching internet services with peer-to-peer interactions. WWW '17. Albert van der Linde et al.

Application example

- Pokemon Go - interactive game
  - Catch Pokemon
  - Pokestops
  - Battles
  - Trading

Client-side: Stronger consistency

- We already replicate logic and state at the client side
  Shared data-structures (maps, lists…): CRDTs, causal consistency
  Supports collaborative applications nicely
  (e.g.: text editors)

- **Has to be done:**
  - Add support for other application requirements, examples:
    - Invariants are hard (e.x.: team pokeballs >= 0)
    - Atomicity (e.x.: all or nothing when trading)

Challenges

- **Has to be done:**
  - Add support for other application requirements, examples:
    - Invariants are hard (e.x.: team pokeballs >= 0)
    - Atomicity (e.x.: all or nothing when trading)

    - **Large amount of writer nodes**
    - **Network and hardware heterogeneity**
    - **Fast paced interactions (e.g.: battles)**

Challenges

- **Has to be done:**
  - Add support for other application requirements, examples:
    - Invariants are hard (e.x.: team pokeballs >= 0)

What about misbehaving users?!

  - **Large amount of writer nodes**
  - **Network and hardware heterogeneity**
  - **Fast paced interactions (e.g.: battles)**

Client-side: Security Mechanisms

- User console commands:
  - Non-permitted actions:
    - ✴ `addPokeballs(‘self’, ‘10.000’)`
  - Networking:
    - ✴ `onMessageFrom(‘enemy_player’) -> drop`
  - Order of events:
    - ✴ `onEvent(lose, time) -> dropEvent & sendMessage(‘leave’, time-5 seconds)`

Client-side:
Security Mechanisms

- We have
  - Access control lists kept by the server; peer-to-peer network removes users that have their access revoked

- **Has to be done:**
  - Create mechanisms to discover and deal (in time) with malicious behaviour

Challenges

- **Has to be done:**
  - Create mechanisms to discover and deal (in time) with malicious behaviour
  - **How to do client-side verification of operations**
  - **How to deal with clients not following protocols**
  - **How to deal with user groups together trying to actively disrupt individual users (many vs few)**

Roadmap

- *What we have*
  - *Legion - Framework to develop interactive web-applications*
    - *Peer-to-peer and shared data-structures (lists, maps…)*
- *What remains to be done - client side*
  - *Support stronger consistency and deal with misbehaving users*

  - Server side - assist client side
    - Distinguish between what clients can do by themselves and what requires assistance from a trusted component
    - Bring closer to end-users: edge
      - Partition logic and state to nearby end users
      - Secure (trusted) computations nearby users

Edge-Cloud hybrid Model for Distributed Applications

Albert van der Linde
a.linde@campus.fct.unl.pt
NOVA LINCS - DI, FCT; Universidade NOVA de Lisboa