Decentralized Computations

Michał Król <m.krol@ucl.ac.uk>, Ioannis Psaras <i.psaras@ucl.ac.uk>

University College London

Decentralized vs Cloud computing

- Low delay
- Lower bandwidth usage
- Lower energy consumption
- Low cost

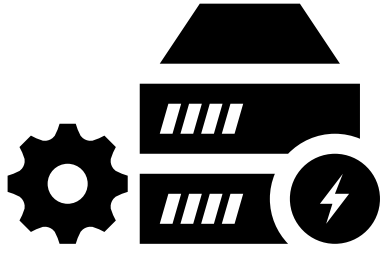
- Untrusted nodes
- Trust in big companies
- Privacy issues

Decentralized computing

To make decentralized computing a realistic alternative:

- Security **and** privacy must be built in the system design
- It must be easy to join the network to submit/execute tasks/repartition tasks
- Nodes need to be rewarded for their work
- Fully decentralized without "trusted" 3rd parties

Scenario



Building blocks

- Rewards
- Result Verification
- Tasks dispatching
- Privacy

Rewards

- Nodes need to be rewarded for used resources
- It can be the main motivation for nodes to join
- Work need to be proved/verified before payment
- But when should be the payments done?

Result Verification

- Different types of tasks
- Cryptographic proof
  - High cost
  - Not available for every computation
- Parallel execution
  - Partial or complete
  - Highly innefficient
  - How to prevent colluding?

Task Dispatching

- How to connect requestors and execution nodes?
- Advertise tasks vs node capacities
- High volume of advertised data
- For delay-sensitive tasks a DNS-like system is impossible
- If rewards are involved, the system must be fair

Privacy

- Called functions
- Input Parameters
- Result Data

All of the above should be hidden from the network **and** from the execution node

Calls Privacy

- If we advertise, the calls become public
- Usage of pseudonyms does not solve the problem
- Calls privacy requires a proxy (zerocash)

Input/Result Privacy

- Homomorphic encryption
  - Introduces overhead
  - Not always possible
- Trusted Execution Environment
  - Creates a trusted environment within an untrusted node
  - Low overhead
  - Requires dedicated hardware

Industry

- Golem, Somn
- Run on Ethereum Blockchain
- Payments using smart contracts
- No automatic, reliable result verification mechanism
- 3rd parties to resolve conflicts

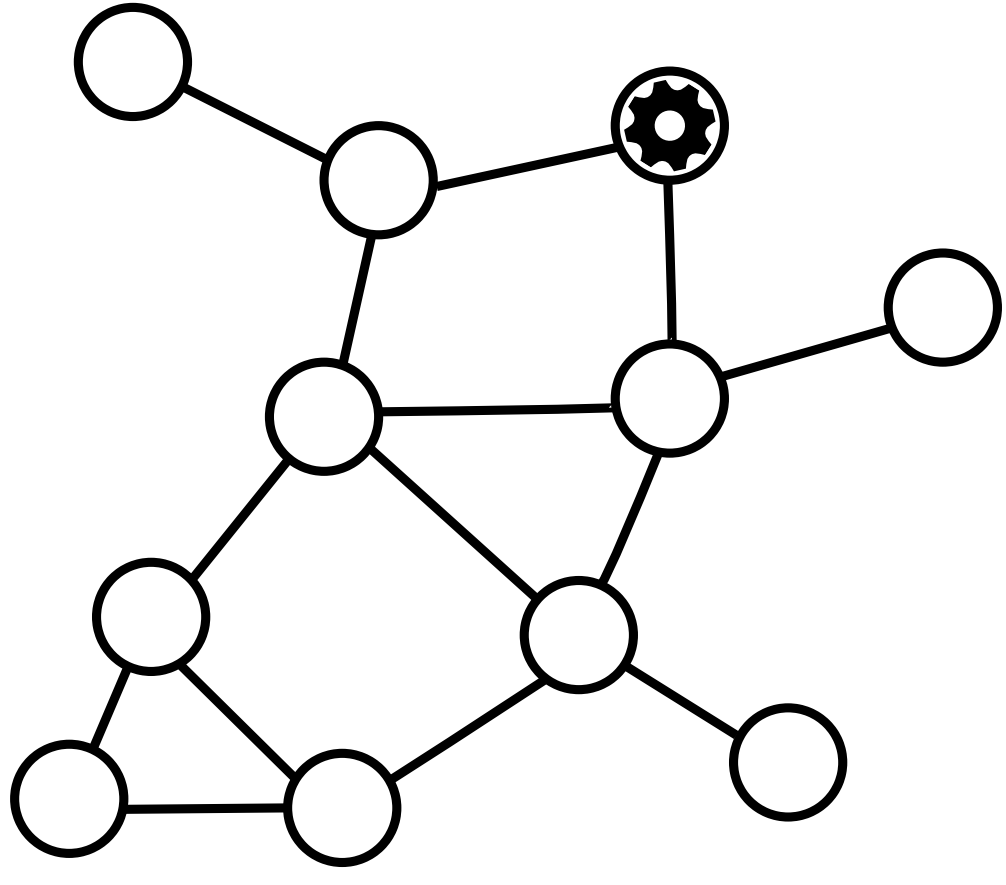
NFaaS

- Rewards
- Result Verification
- Tasks dispatching
- Privacy

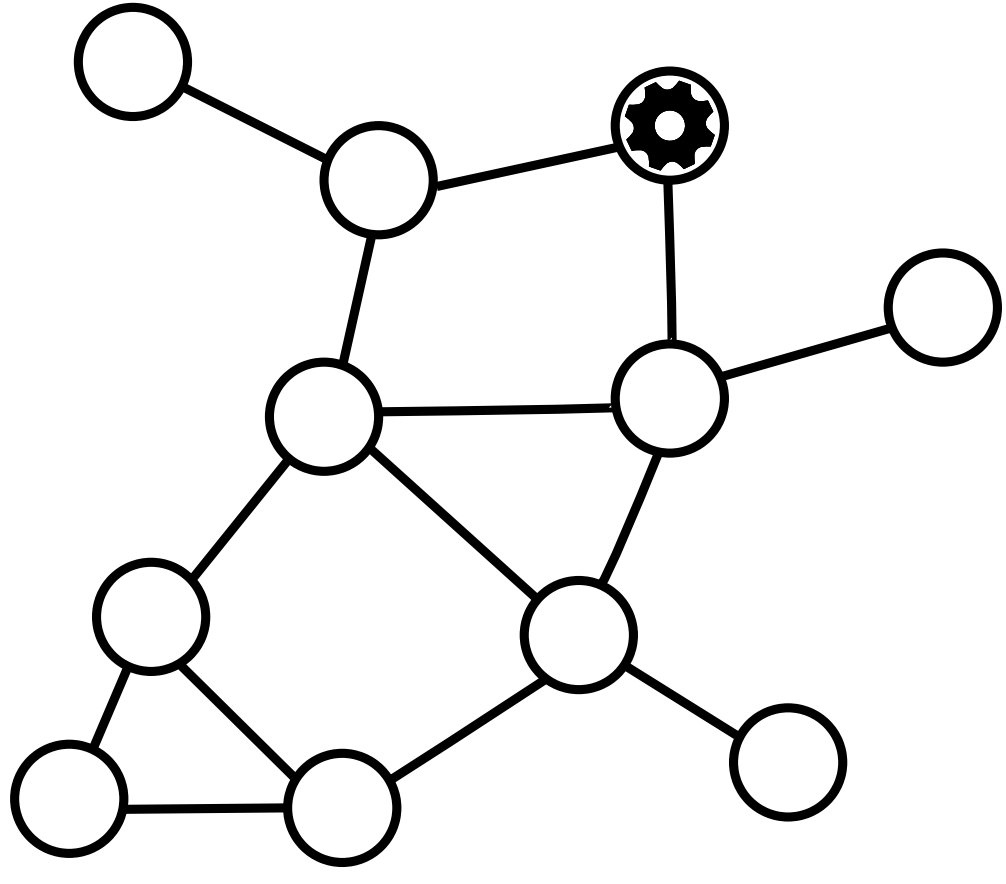
NFaaS

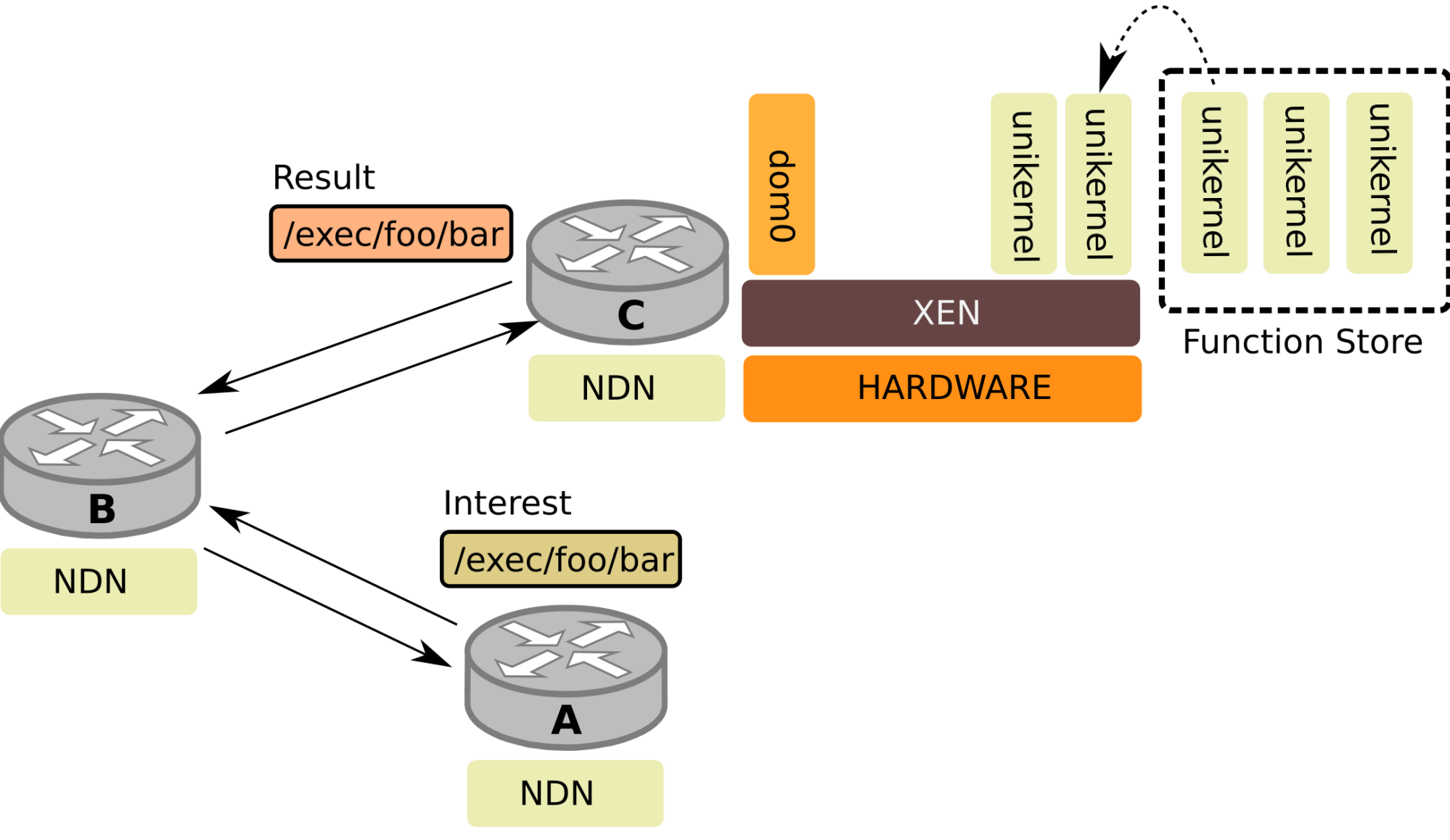
- Task dispatching environment for delay-sensitive tasks
- Function represented as stateless unikernels
- Implemented on top of NDN
- Nodes quickly addapt to network dynamics
- Fully decentralized
- Encrypted communication with functions

Function Execution

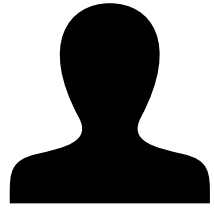


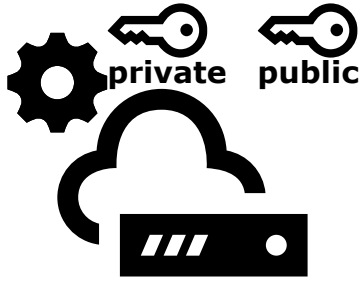
Named Function as a Service

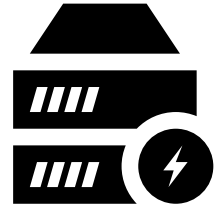



Result
/exec/foo/bar
C
NDN
dom0
unikernel
unikernel
XEN
HARDWARE
unikernel
unikernel
unikernel
Function Store
B
NDN
Interest
/exec/foo/bar
A
NDN

Encryption







NFaaS

- Quickly adapts to network needs
- Small function size based on rumprun
- Fast boot time

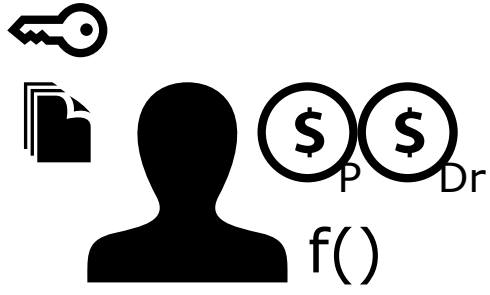
SPOC

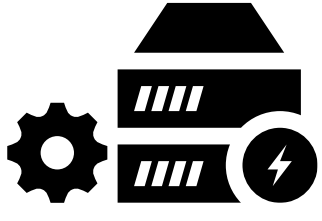
- Rewards
- Result Verification
- Tasks dispatching
- Privacy

SPOC

- Automatic payments and result verification
- Based on Smart Contracts and Intel SGX
- No 3rd parties involved

SPOC





SPOC

- Secure against Rational Attacker
- Minimal computational overhead
- No calls privacy

Thank you