ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Forward-secrecy on POP

Arthur Villard

School of Computer and Communication Sciences

Decentralized and Distributed Systems lab

Master Thesis – September 2017

Responsible
Prof. Bryan Ford
EPFL / DEDIS

Prof. Ewa Syta
Trinity College

Supervisor
Linus Gasser
EPFL / DEDIS

Context

- Online collaborative service (e.g. Wikipedia)
- Authenticate users anonymously against a list
- Link authentication attempts
- Other example: e-voting

Overview

- Introduction
- PoP and DAGA interaction
- Implementing DAGA
- Improving DAGA
- Conclusion & Future work

Overview

- **Introduction**
- **PoP and DAGA interaction**
- **Implementing DAGA**
- **Improving DAGA**
- **Conclusion & Future work**

Frameworks

- **PoP: Proof of Personhood – DEDIS**
  - Creation of the user list
  - Authentication protocol
  - Anonymity within the group
  - No forward-secrecy
- **DAGA: Deniable Anonymous Group Authentication – Ewa Syta**
  - Authentication protocol
  - Forward-secrecy

Goals

- **Using DAGA as PoP's authentication protocol**
- **Implementing DAGA in Go**
- **Improving DAGA**

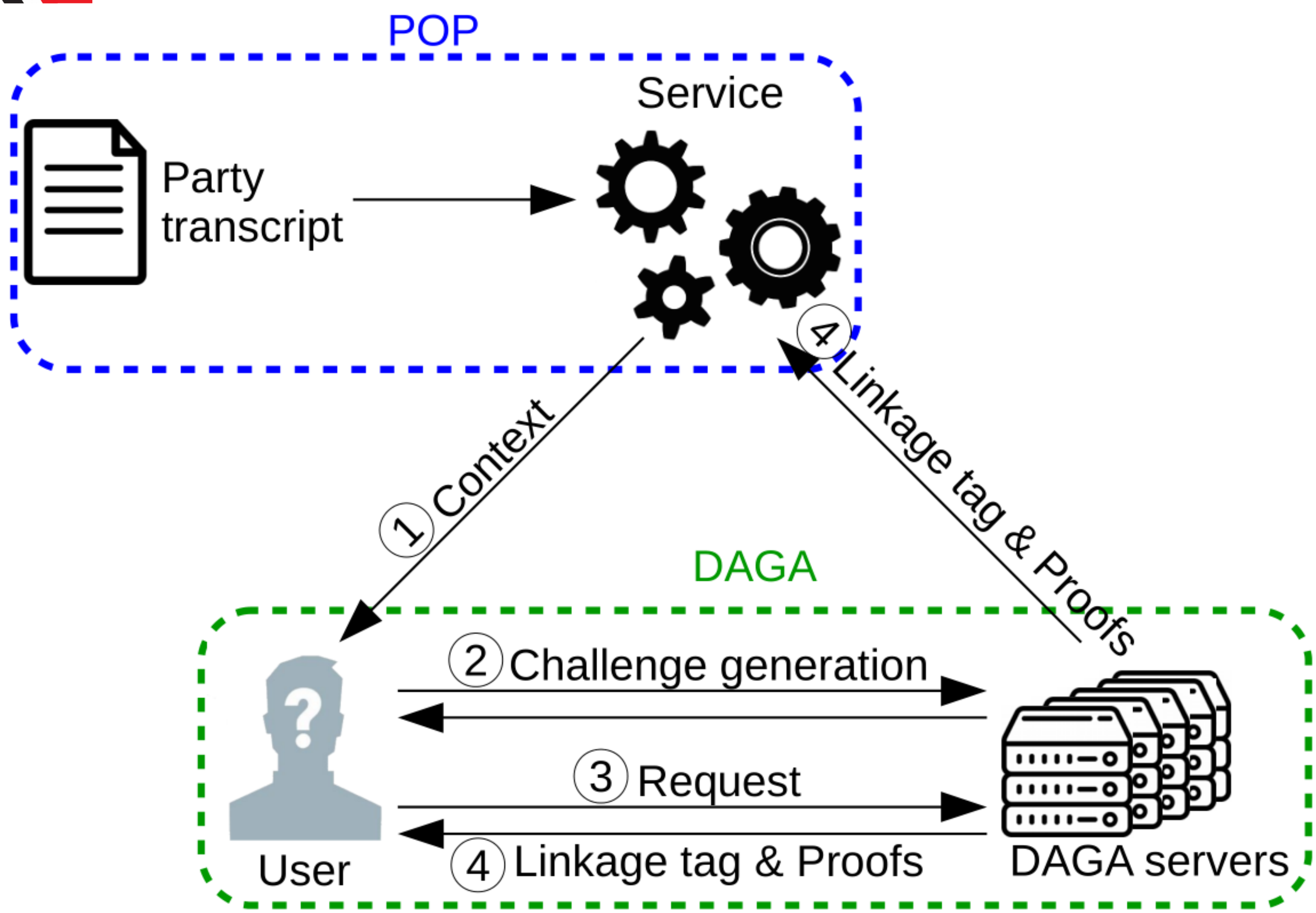
Key concepts

- **Anonymity**
  - ➔ No information about the user is known
- **Accountability**
  - ➔ The sender can be held responsible for his action
- **Linkability**
  - ➔ Two messages come from the same user
- **Forward-secrecy**
  - ➔ Breaking a session does not break the previous ones

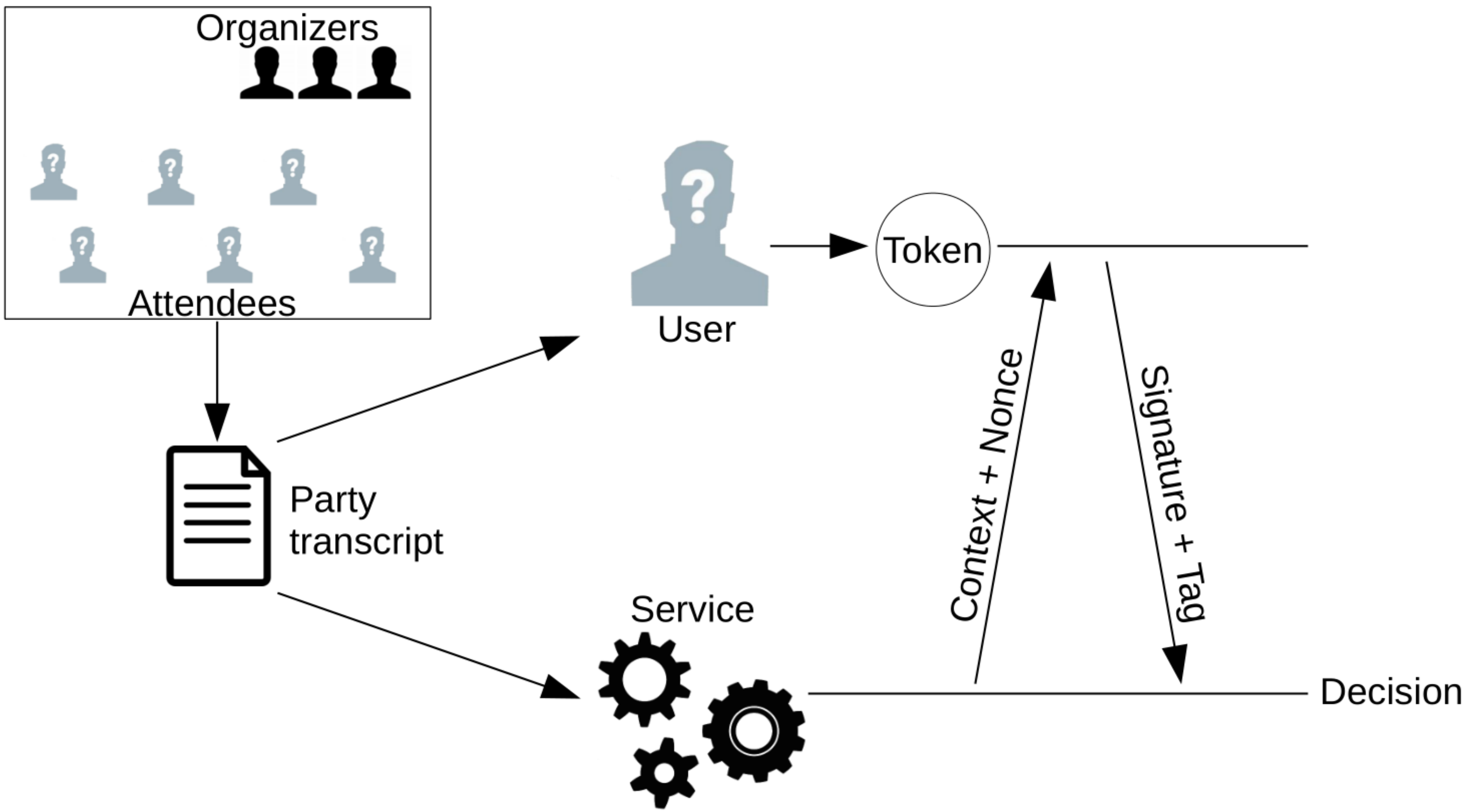
Overview

- Introduction
- **PoP and DAGA interaction**
- Implementing DAGA
- Improving DAGA
- Conclusion & Future work

Integration



PoP: How it works

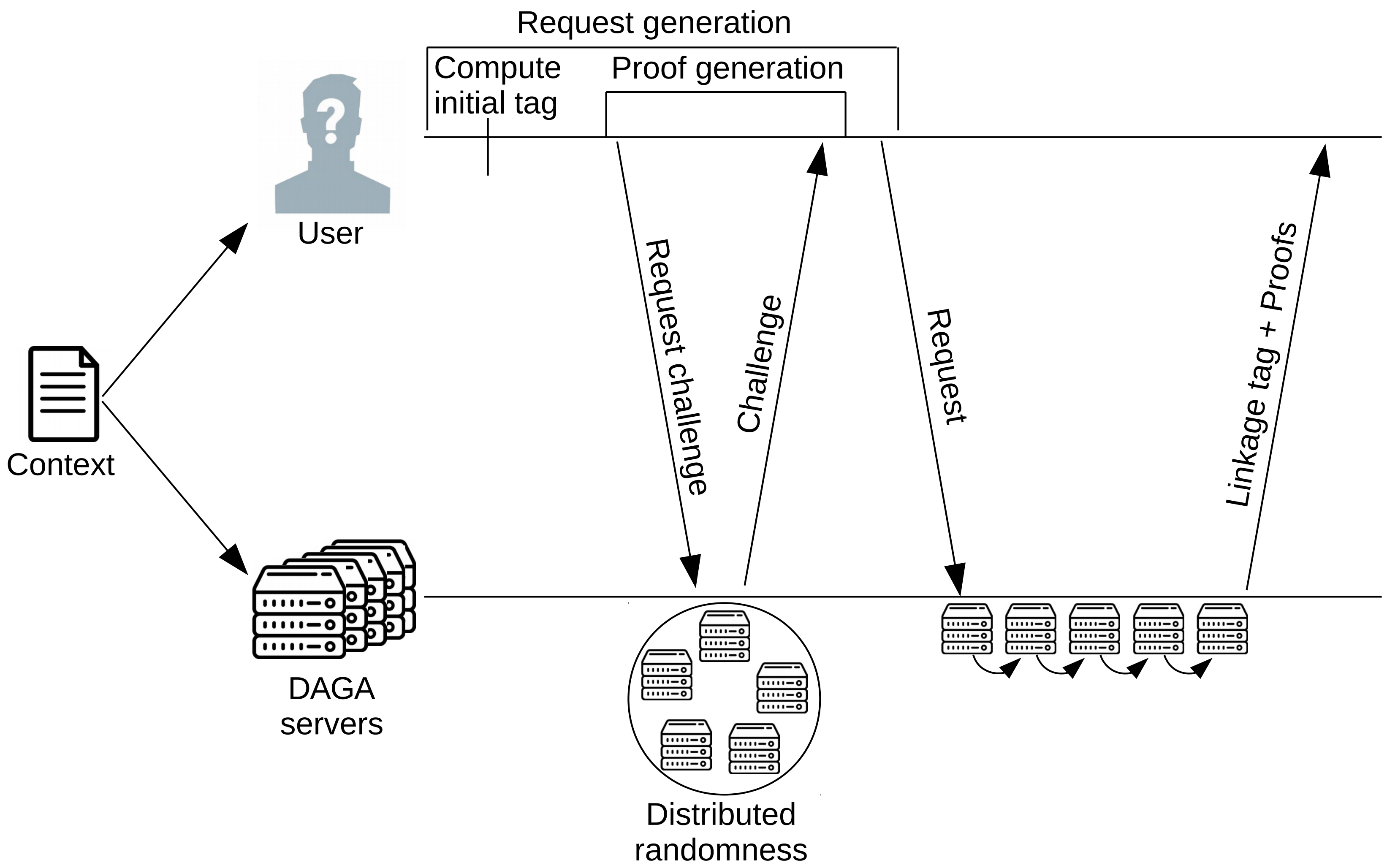


PoP: Weaknesses

- **No forward-secrecy**
  - ➔ Tag derived from private key
  - ➔ Leakage allows to identify the user in previous sessions
- **Cross-service de-anonymisation**
  - ➔ Tags independent from the service
  - ➔ Users can be tracked between different services

→ Loss of anonymity

DAGA: How it works



DAGA solutions

- **Forward-secrecy**
  - Tags derived from context elements only
  - Private key used in client proof
  - Proof does not leak information
- **Cross-service de-anonymisation**
  - Different services → Different contexts

    [?] Different tags for the same user

Conclusion

- **DAGA can solve PoP weaknesses**
- **DAGA and PoP can be interfaced**
- **E-voting**

Overview

- Introduction
- PoP and DAGA interaction
- **Implementing DAGA**
- Improving DAGA
- Conclusion & Future work

Implementation

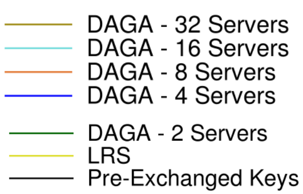
- **Go**
- **RSA $\boxed{?}$ Elliptic Curves**
- **Distributed randomness**

$$T_0^i = h_i^{(\prod_{k=1}^{m} s_k)} \quad \boxed{?} \quad T_0^i = \left(\prod_{k=1}^{m} s_k\right) * H_i$$

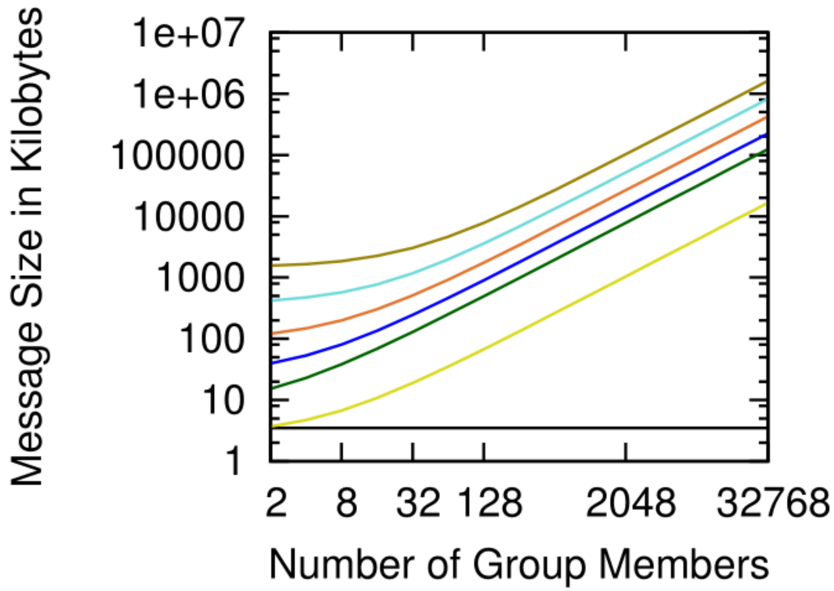
Code results

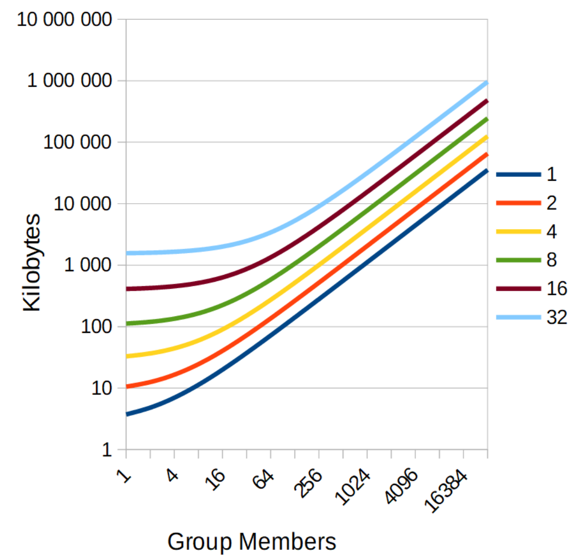
- **Library: Complete implementation**
- **Test coverage 88%**
- **Example scenario**
- **Benchmark package**

Benchmarks: Communication



Setup:
- **Ubuntu 12.04**
- **x86-64**
- **1 thread**

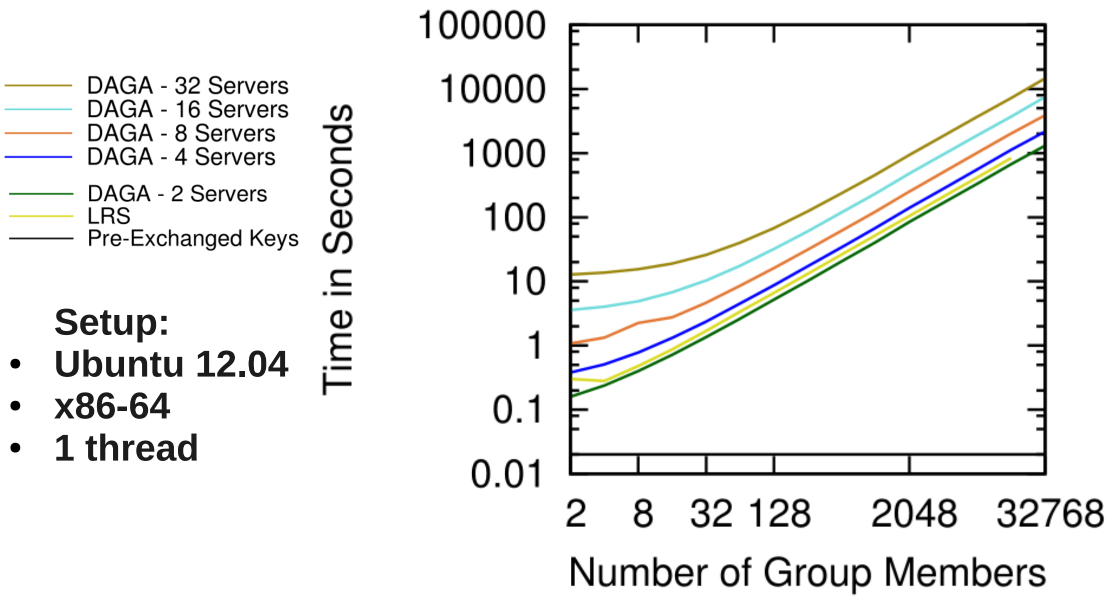




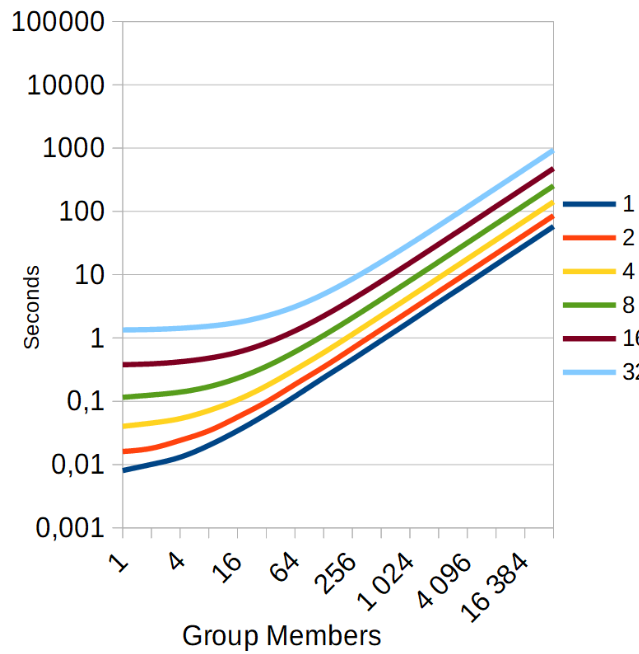
Setup:
- **Windows 10**
- **x86-64**
- **1 thread @4,5GHz**

- **No improvement**
- **No explanation yet**

Benchmarks: Time



Setup:
- **Ubuntu 12.04**
- **x86-64**
- **1 thread**



Setup:
- **Windows 10**
- **x86-64**
- **1 thread @4,5GHz**

32768 members / 32 servers 15 000 s → /15 → 1 000 s

- **Moore's law 2012 → 2018: ~ /8 from hardware**
- **Elliptic Curves**

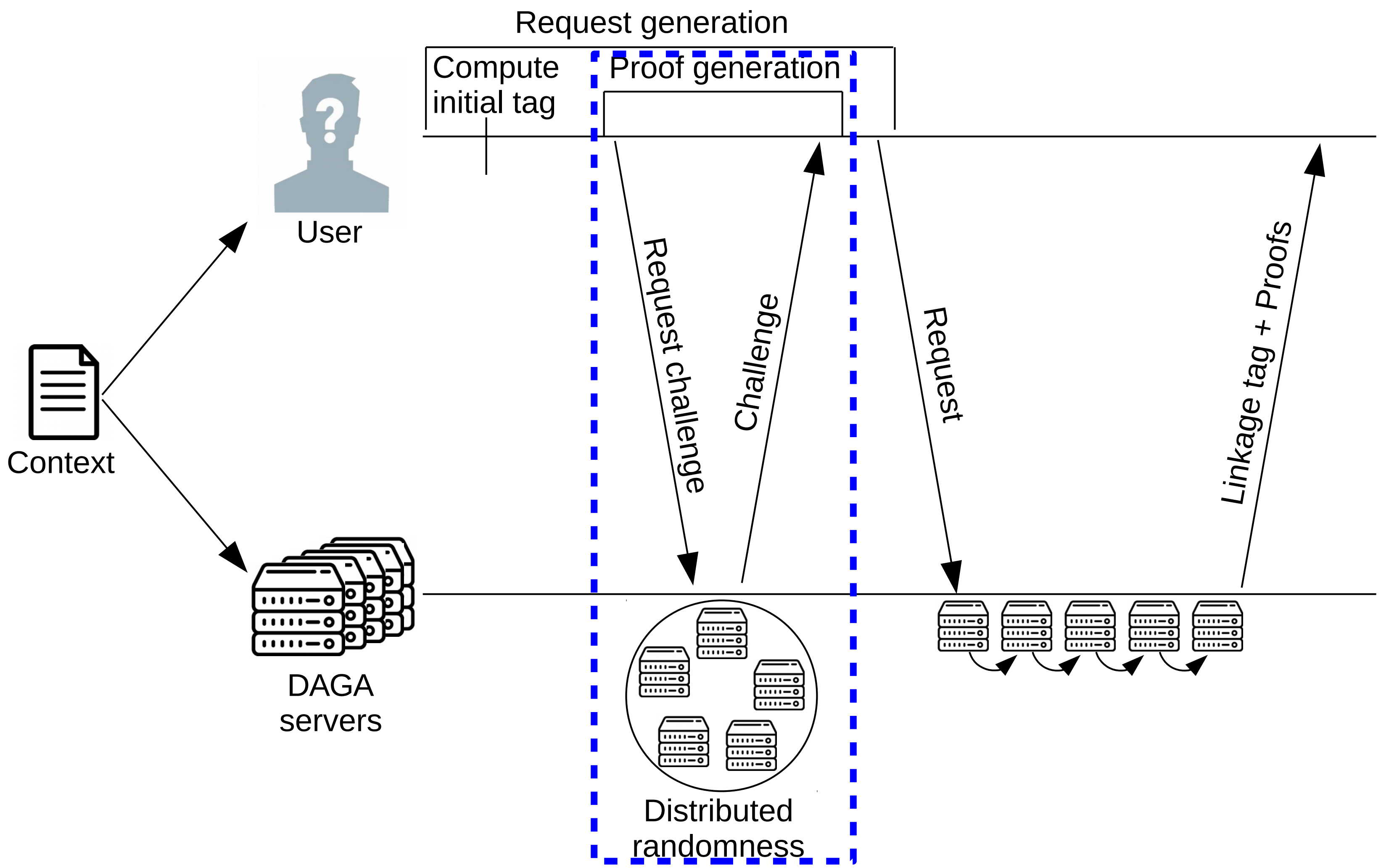
Conclusion

- Complete implementation
- Time improvement
- Next step: Integrate it with PoP

Overview

- Introduction
- PoP and DAGA interaction
- Implementing DAGA
- **Improving DAGA**
- Conclusion & Future work

Proof problem



Proof problem

- **Anonymity through a client OR proof:**
  - ➔ I know (private key 1 OR private key 2 OR …)
- **Growth $O(6*n)$, n = #members**
  - ➔ 32768 members / 32 servers
    - Proof ~6,3 MB, total cost ~200 MB → ~20% of total

Improving the proof

- **Work with Kasra Edalatnejadkhamene, PhD student**
- **Survey of the field**
- **Split the proof**
  - → Proof of membership: Accumulator
  - → Proof of knowledge: Signature of knowledge
- **No concrete scheme**

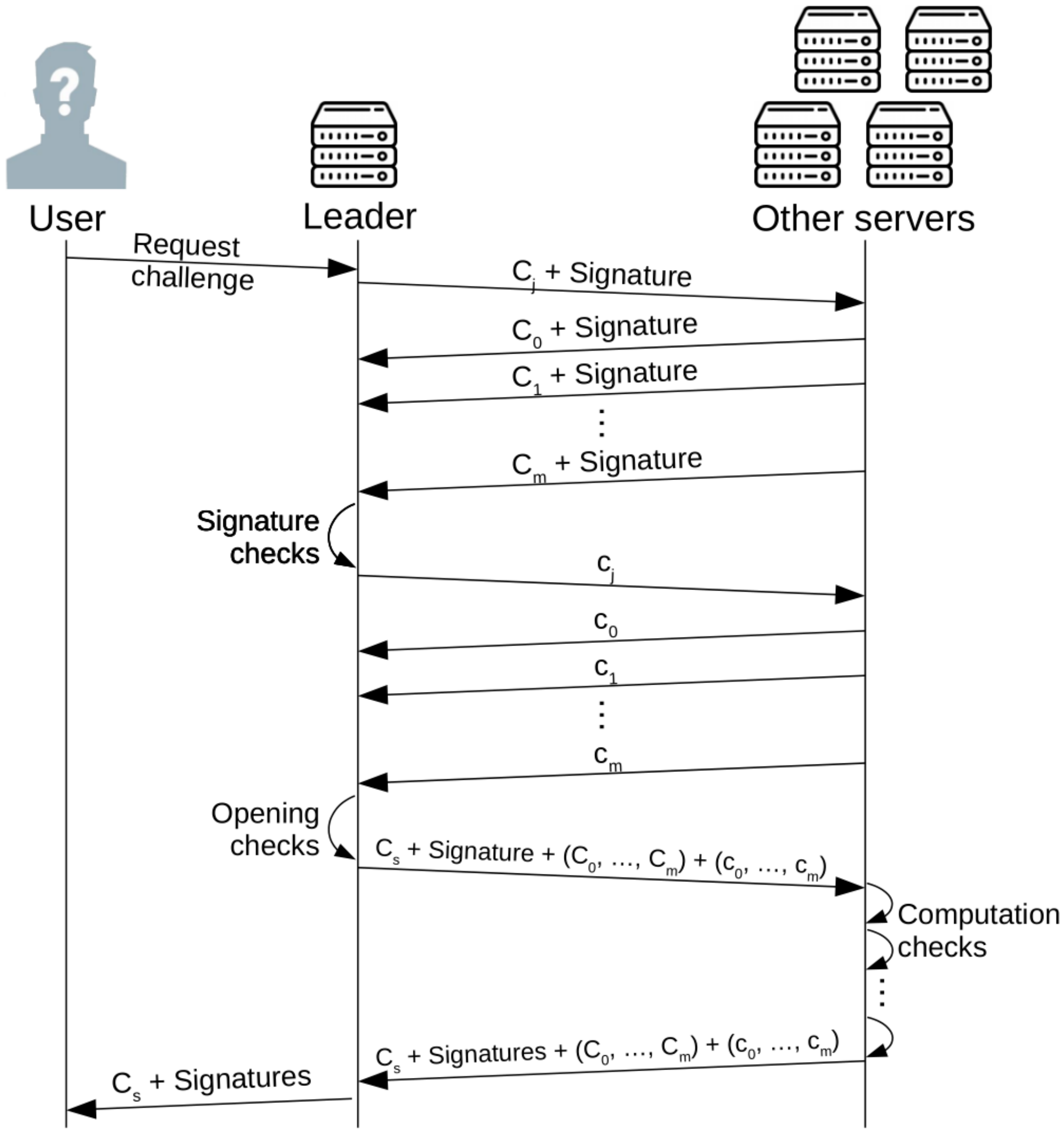
Overview

- Introduction
- PoP and DAGA interaction
- Implementing DAGA
- Improving DAGA
- **Conclusion & Future work**

Conclusion & Future work

- **DAGA and PoP can work together**
- **Complete Go implementation of DAGA**
- **Improvement guidelines for the proof**
- **Next steps**
  - → Integrate DAGA and PoP
  - → Optimize network consumption
  - → Continue the work on the proof
  - → Improve implementation resistance (secure memory management, constant-time, …)

Distributed randomness



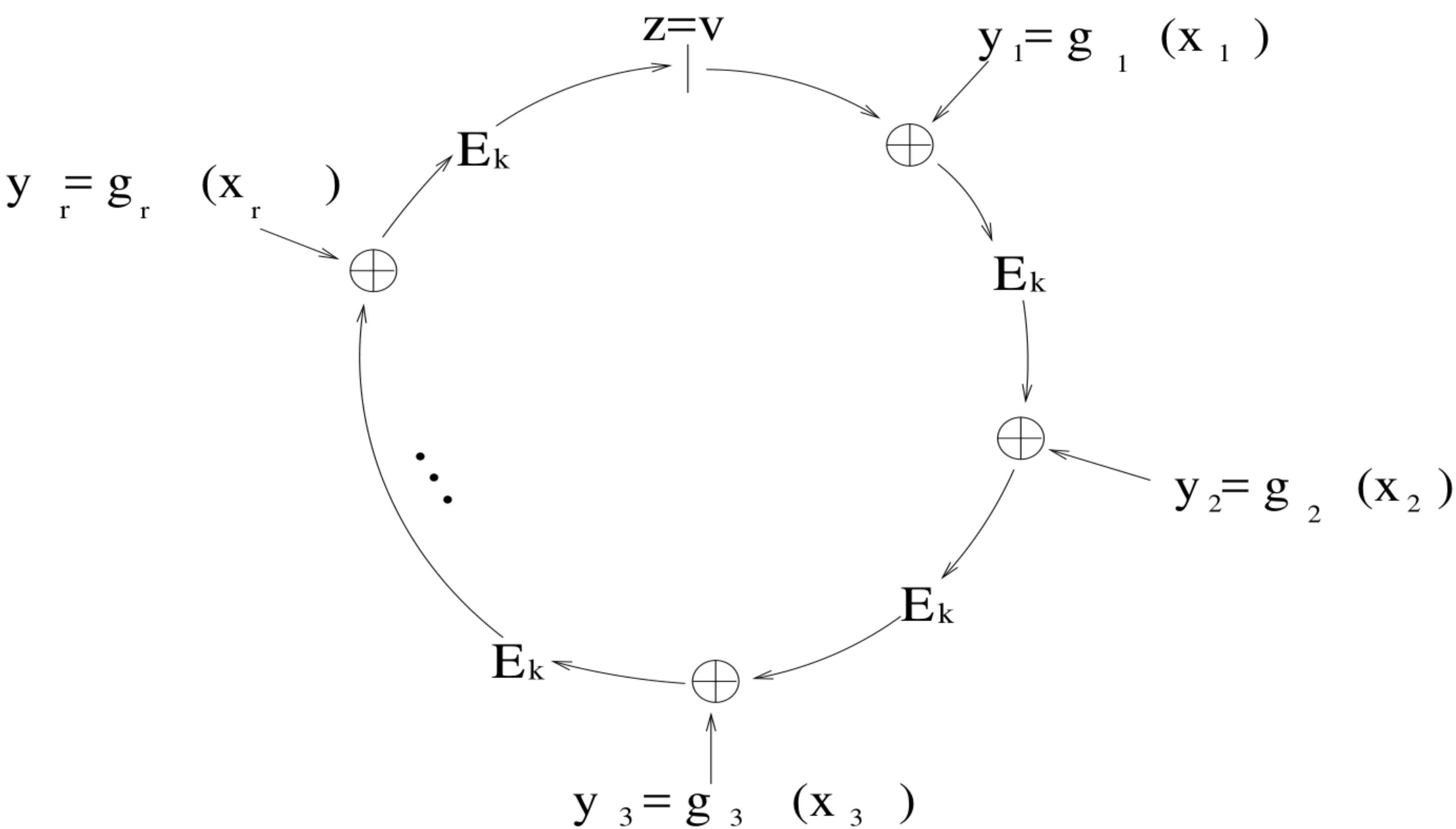
Context

- User public keys (#members)
- Server public keys (#servers)
- Server random commitments (#servers)
- Client random generators (#members)

Accumulator

- ***Accumulators from Bilinear Pairings and Applications***
  L. Nguyen, 2005
- **Adjustments:**
  - Trusted setup
  - Bounded
  - Efficiency based on trusted authority

Ring signature



- *How to Leak a Secret*, R. Rivest, A. Shamir and Y. Tauman