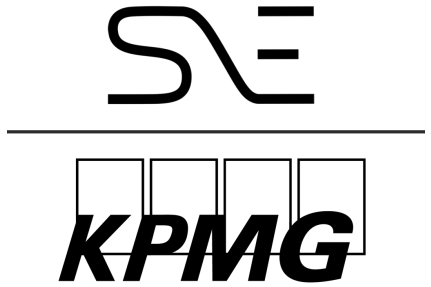
UNIVERSITY OF AMSTERDAM

Signing containers images with Docker Notary

Mohanad Elamin
University of Amsterdam
melamin@os3.nl

Rio Kierkels
University of Amsterdam
rkierkels@os3.nl



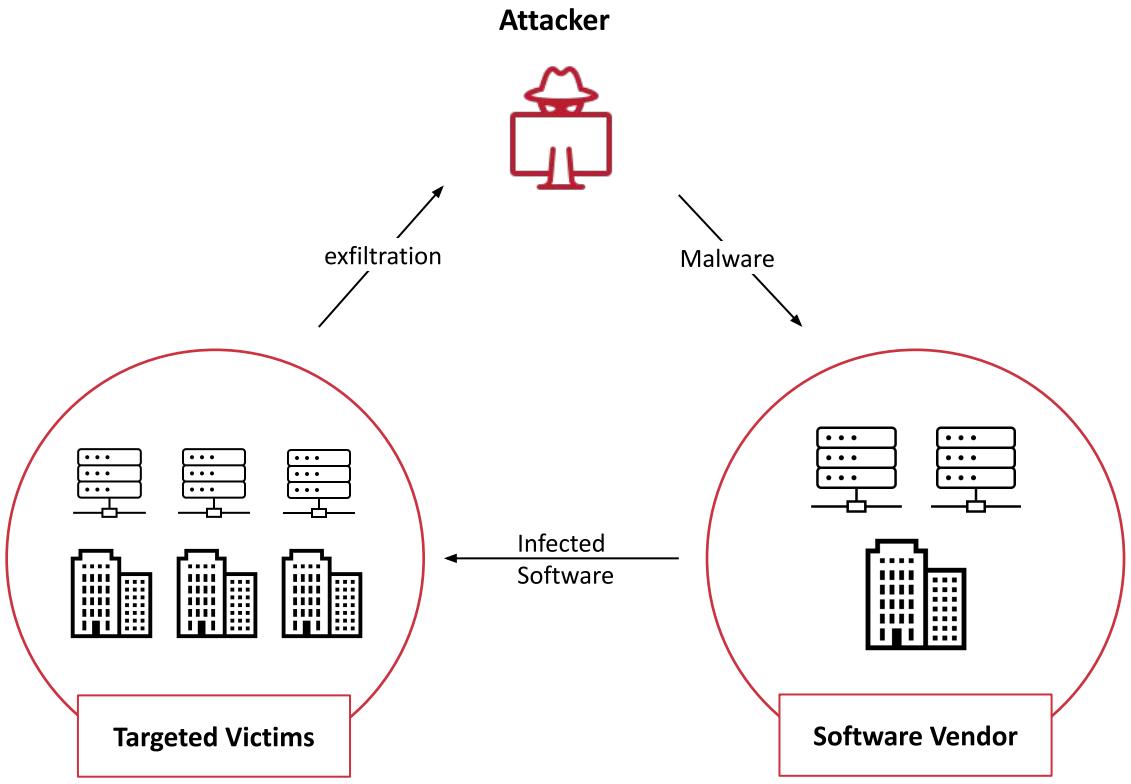
Supervisors:

Aristide Bouix
KPMG

Jasper Boot
KPMG

Introduction

- **Content Trust** is a fundamental security concern for software update system.
- **The Update Framework** is a framework for securing software updates.
- Docker **Notary** is a Go language implementation of TUF.

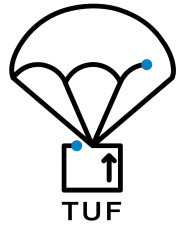


Research Question

What are the best practices of using Notary for container image signing?

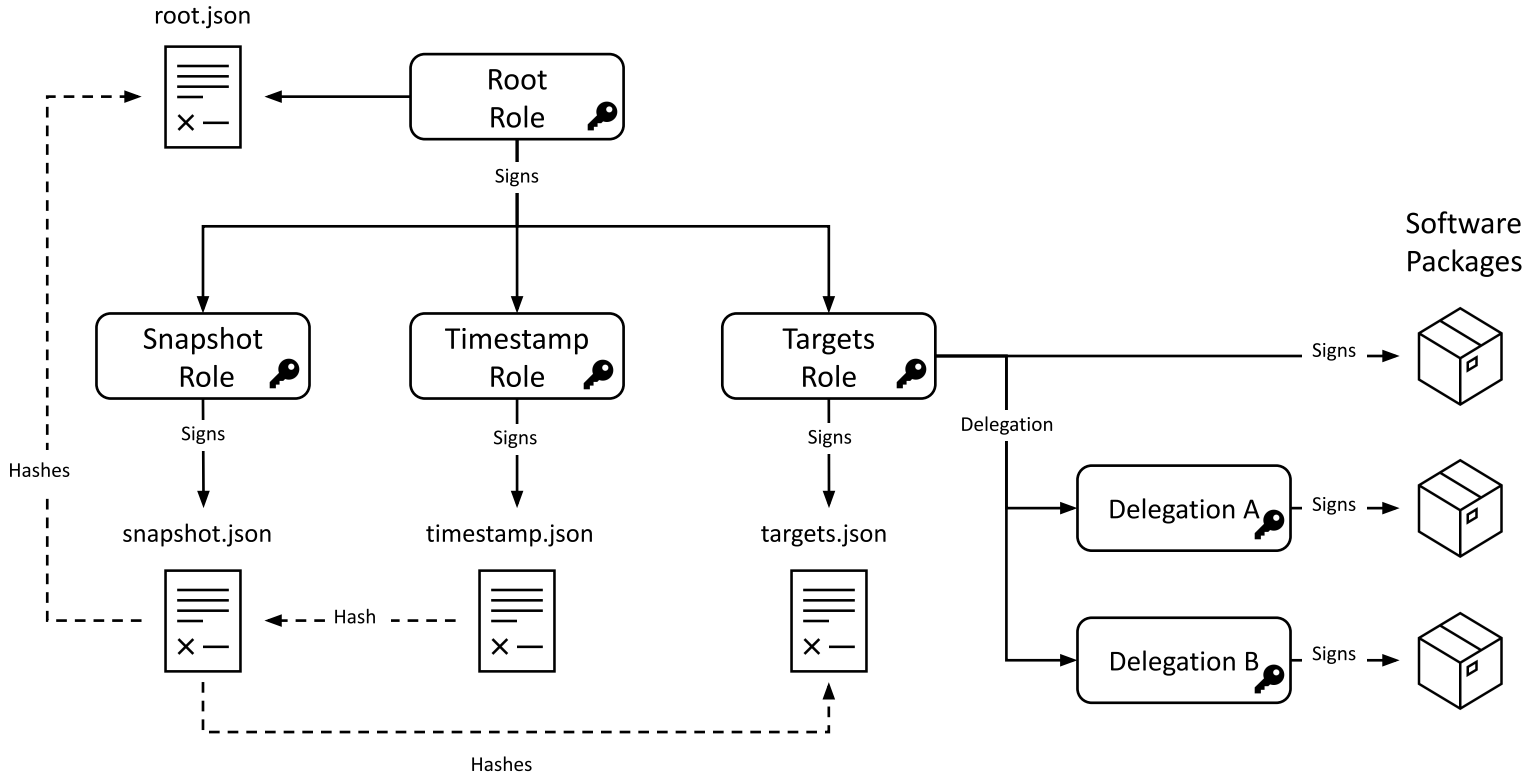
- How does Notary ensure the integrity and security of container images?
- What are the challenges of deploying Notary for container image signing?
- Based on the Proof of Concept test results, what are the main probable reasons of low adoption for Notary and container image signing?

Background - The Update Framework



Design Principles:

1. Separation of duties.
2. Threshold signatures.
3. Explicit and Implicit revocation of keys.
4. Minimizing Risk.



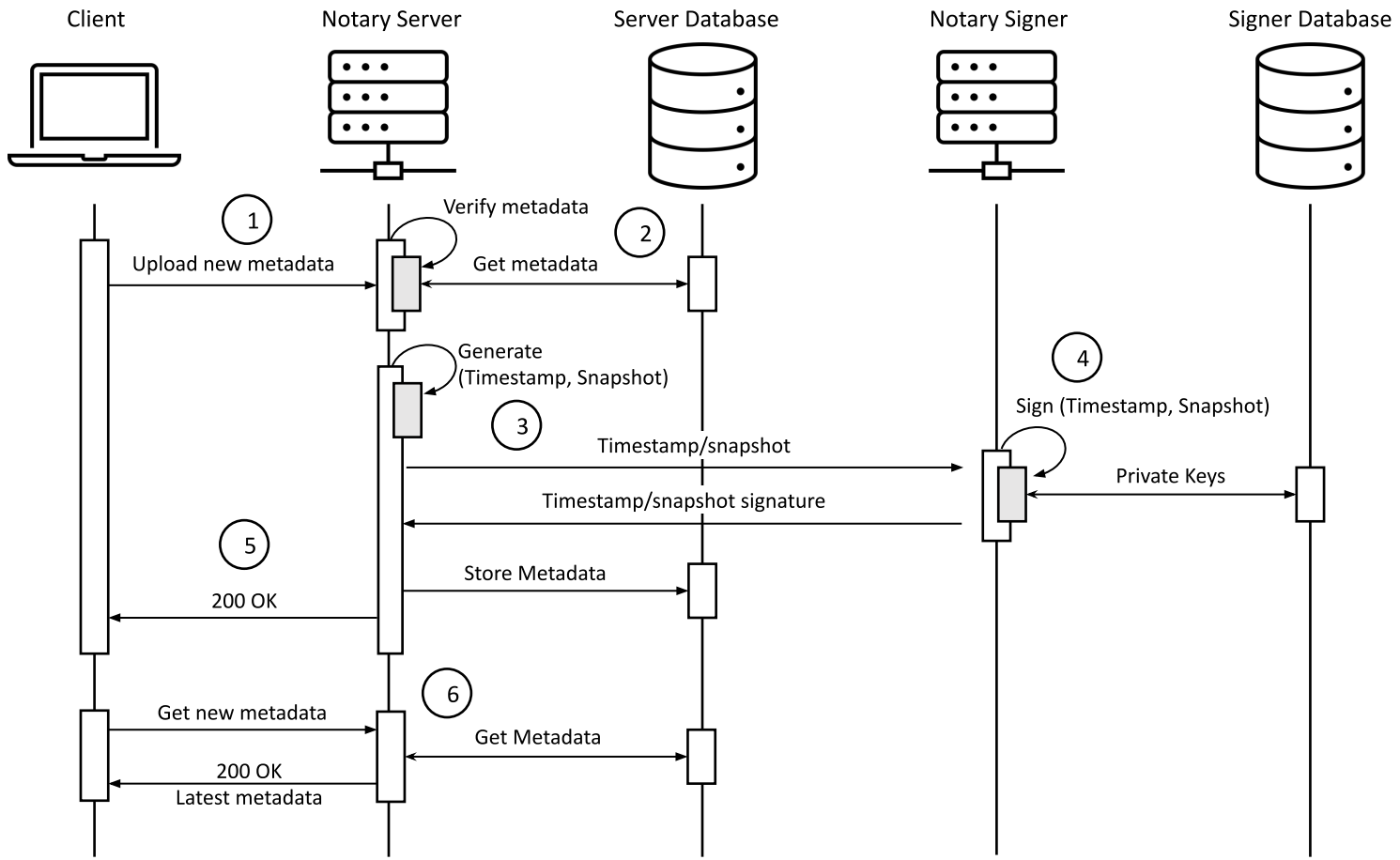
Background - Docker Notary

- **Notary Server:**

  Keeps, updates and ensures TUF metadata validity.

- **Notary Signer:**

  Sign metadata using the stored Timestamp and Snapshot keys.



Related Work

- Security Assurance analysis of Docker containers from the DevOps model's angle.
- Multiple Penetration tests on Docker Notary and The Update Framework.
- Some highlights from Commercial enterprises.

Methodology

- Create manifests that encode best practices for deploying Notary.
  - Good starting manifests for production usage.
  - Clear separation of core Notary and it's dependencies.
  - Swappable components.
  - TLS everywhere.
- Build experience about deploying the system.
- Build and understanding of its daily operations.

Happy Path Demo

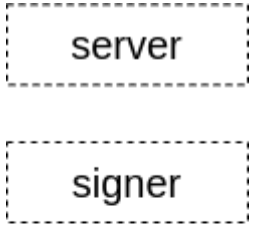
https://www.youtube.com/watch?v=jcgkMYyzeYY

Methodology

Day 0 - Design

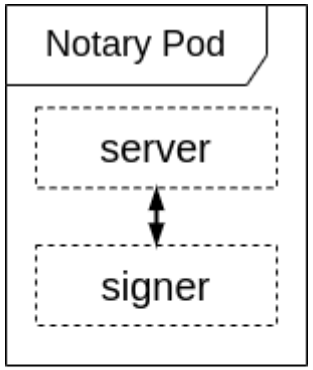
Methodology

Day 0 - Design



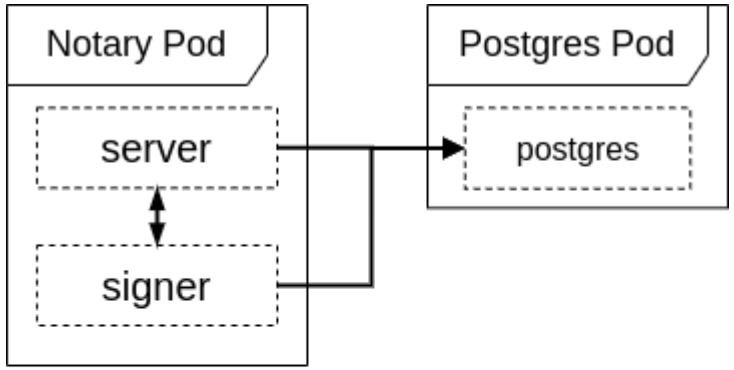
Methodology

Day 0 - Design



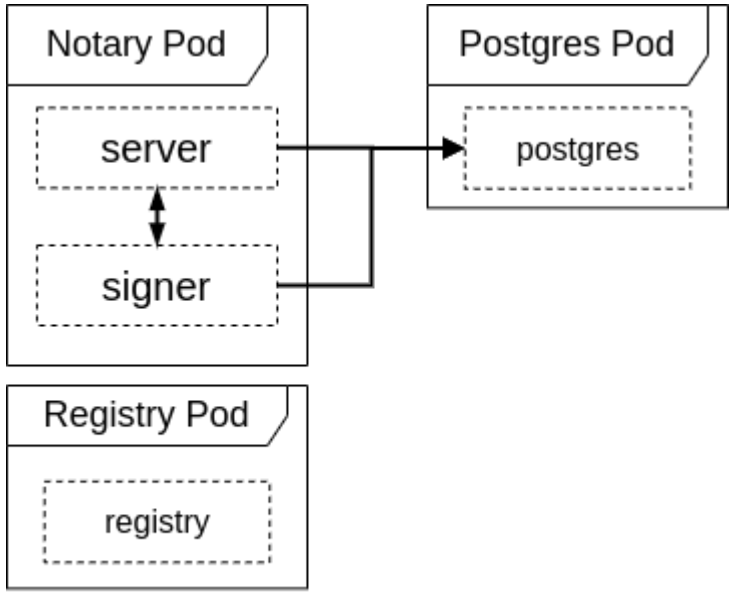
Methodology

Day 0 - Design



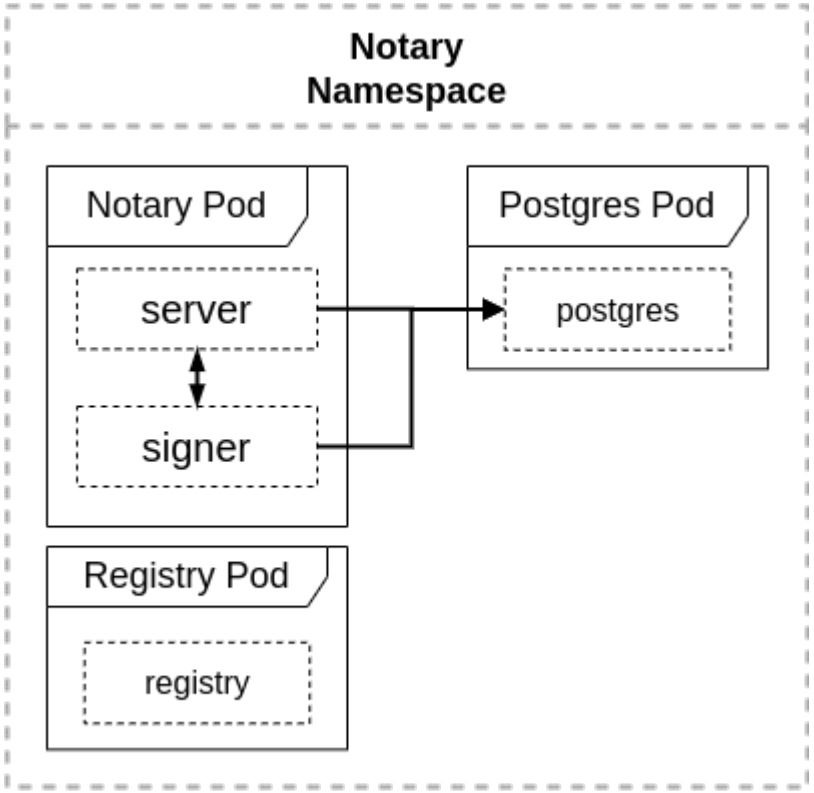
Methodology

Day 0 - Design



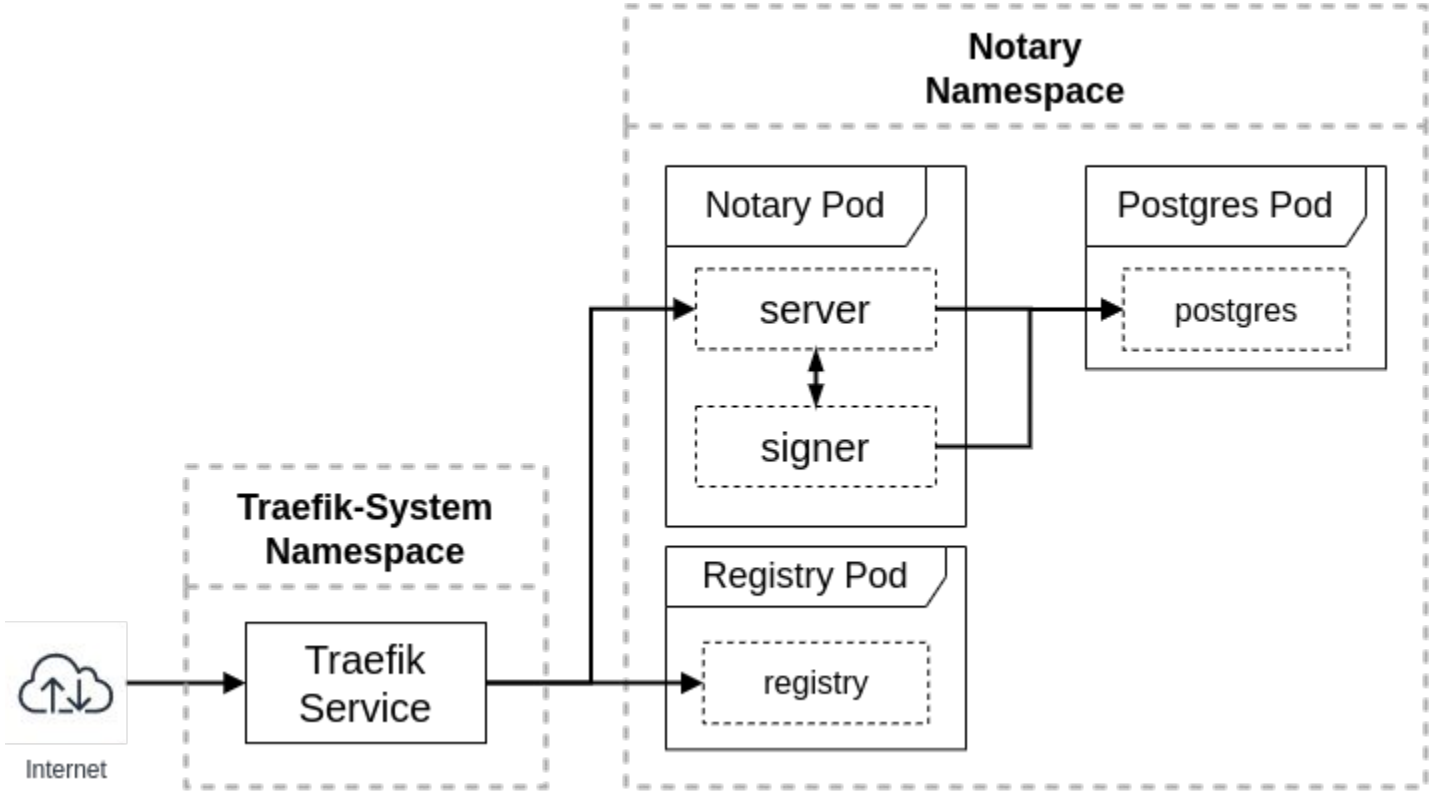
Methodology

Day 0 - Design



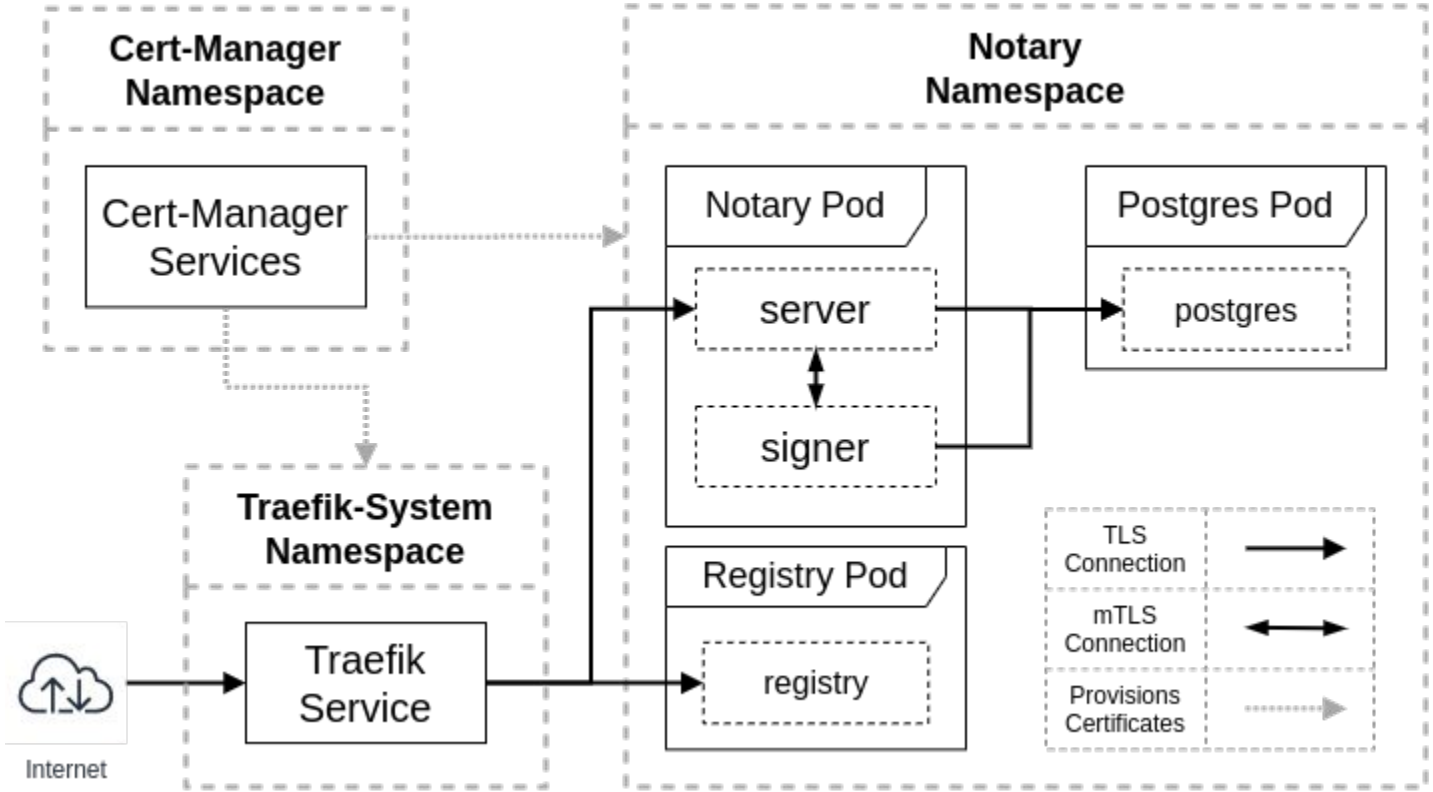
Methodology

Day 0 - Design



Methodology

Day 0 - Design



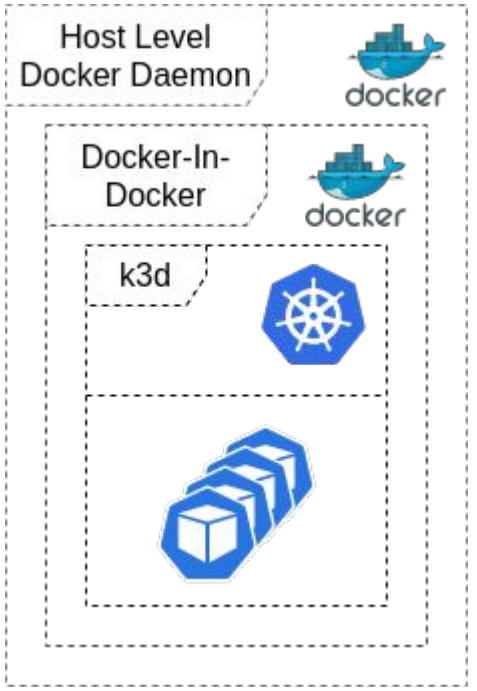
Methodology

Day 1 - Deployment

Methodology

Day 1 - Deployment

- Sandboxed test environment
  - Fast setup and teardown
  - Reproducible
  - System isolation
  - Single dependency



Methodology

Day 2 - Operations

Methodology

Day 2- Operations

- Image tamper detection.
  - Layer and manifest manipulation.
- Target key compromise.
  - Only the compromised delegation can be abused.
  - Requires key revocation and access to the repository key.
- Root key compromise.
  - All keys can be rotated by the attacker.
  - Requires a new root key and access to the old root key.

Conclusion

- Deployment is not straightforward and easily misconfigured.

- Notary and TUF work as advertised but the Notary abstraction is leaky. Docker's usage of Notary is not enforced by Notary.
- Key management and general administration is complicated. Lack of tooling and integration.

- Notary V1 development has stalled, V2 is on the horizon.

Future Work

- External Survey to understand the adoption challenges.
- Research the authentication subsystem of Notary.
- The use of Notary along with other framework like in-toto for holistic software supply chain security.
- Collect feedback about our proposed manifests.

Thank You

Paper + Manifests

https://github.com/rio/notary-kubernetes/

Mohanad Elamin
University of Amsterdam
melamin@os3.nl

Rio Kierkels
University of Amsterdam
rkierkels@os3.nl

Supervisors:

Aristide Bouix
KPMG

Jasper Boot
KPMG