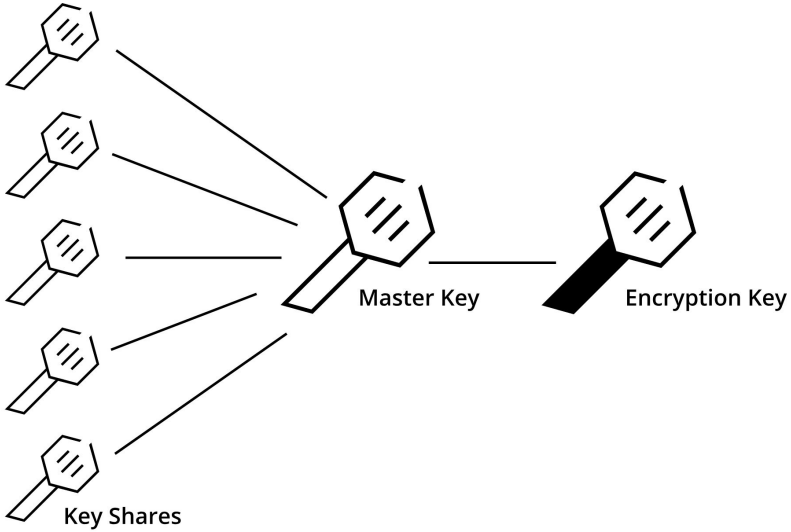


Universität Potsdam

HPI Hasso Plattner Institut
Digital Engineering · Universität Potsdam

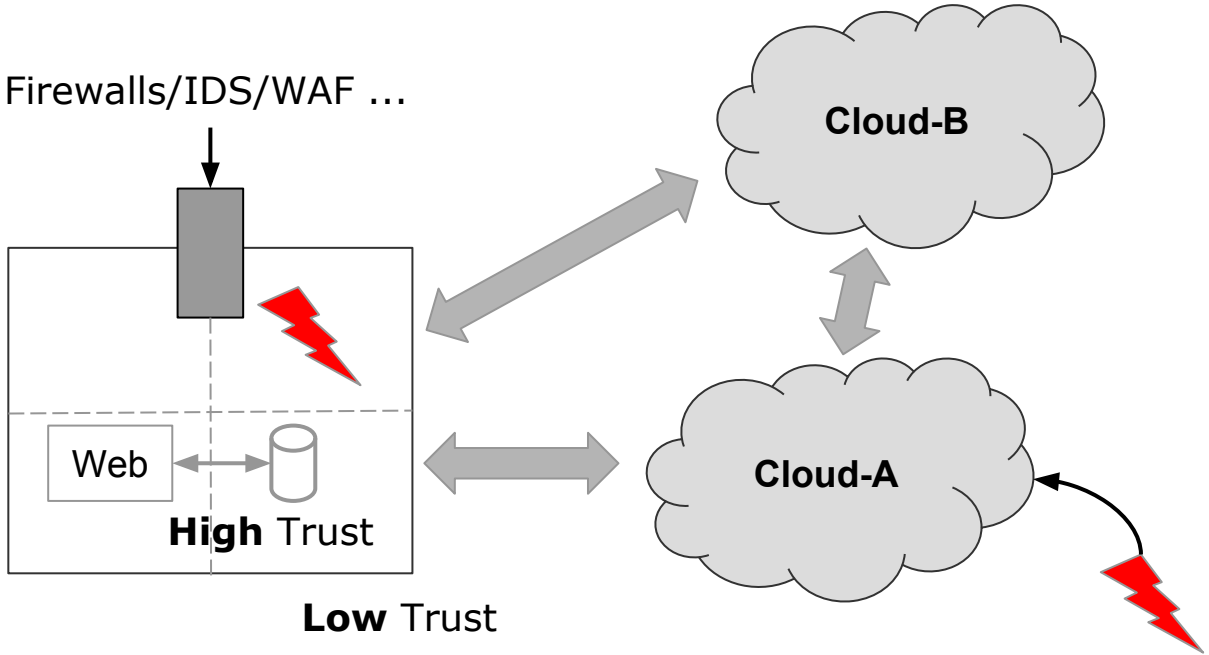
Secrets Management

Johannes Kroschewski
Nils Straßenburg
Network Security In Practice - Final Presentation

Agenda

- Motivation: Why Secrets Management?
- The problem of having a single master key
- How to share and split a secret: Shamir’s Secret Sharing
- Authentication and storing of secrets
- Challenges
- Example
- Future Work and Achievements

Motivation - Multi Cloud Security



Verizon's 2018 Data Breach Investigations Report

Secrets Management - Definition

- Distribution of secrets to the end application?
  - e.g.: server needs access to DB
- Update secrets?
  - e.g.: credentials should not last forever
- Revoke secrets?
  - e.g.: manage the case that a system is compromised
- Define a **life cycle** of a secret ...

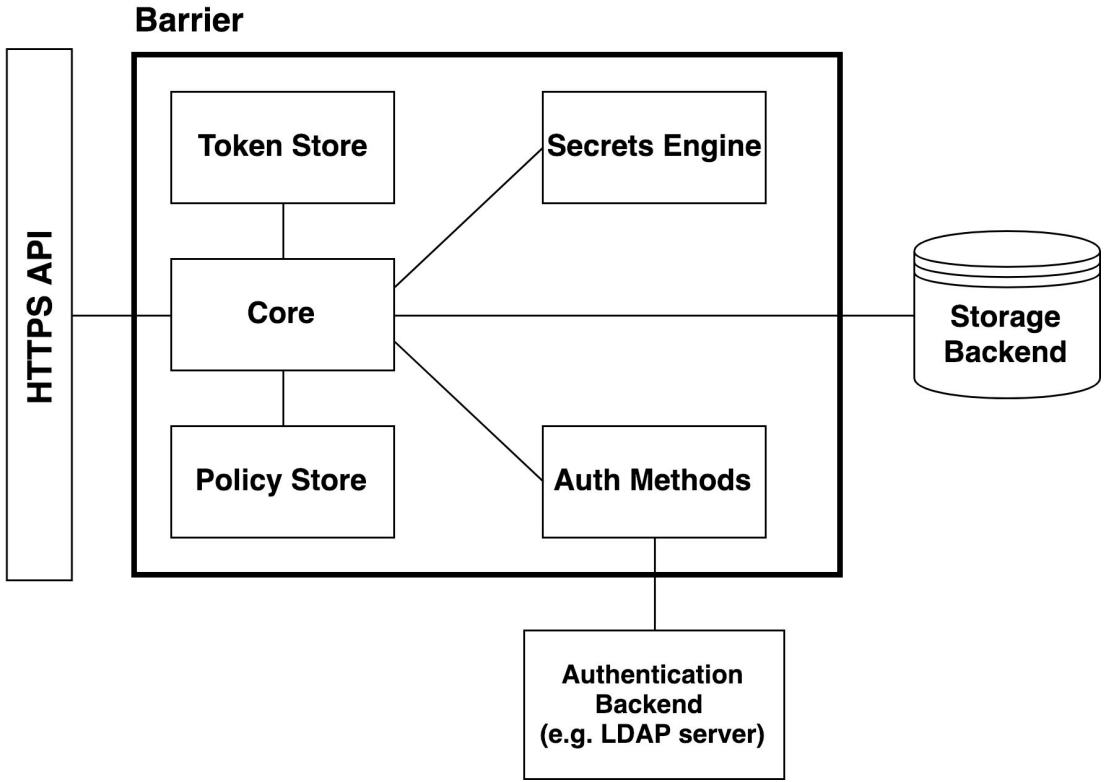
Building a process around these points is Secrets Management

Secrets Management - Use Cases

Vault: Use Cases

- General Secret Storage (*vault read*)
  - storage of any secrets (e.g. env variables, database credentials)
- Employee Credential Storage (overlaps with GSS)
  - share credentials between employees
  - audit logs
- API Key Generation for Scripts - *dynamic secrets*
  - generate secrets only for duration of a script
- Data Encryption
  - use vault to encrypt/decrypt data elsewhere

Secrets Management - Example - Vault



Vault Architecture

Encrypting Secret Information - Problems

- For Secrets Management one instance is introduced to manage and store all secrets
- Information is encrypted with some key (e.g. AES)
- Who should hold the key?
- Should a **single** user hold this key?
  - Holder of key passes away
  - The key is compromised via a malicious hacker
  - The holder of the key turns rogue

Encrypting Secret Information - Solutions

- Should a **single** user hold this key?
  - Holder of Key passes away

    Store the encryption key on the (SM) system

    ok we can do that
  - The key is compromised via a malicious hacker
  - The holder of the key turns rogue

    “Two-man rule”: Make sure that no user can access the encryption key without the agreement of other users

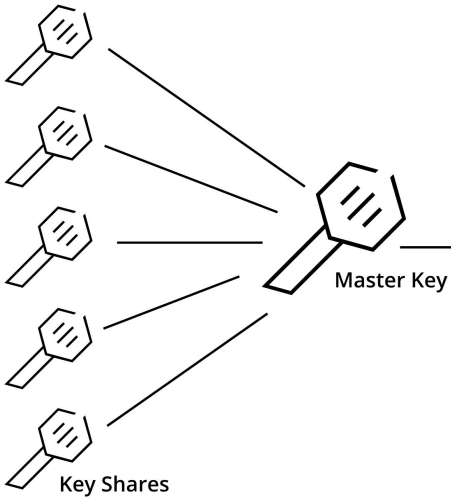
    ok let’s do it

    but how ?!

Securing access to encryption key

- Encryption key is stored on management instance
  - encrypt encryption key with another key (*master key*)
- No user should be able to encrypt the encryption key alone
  - **split master key** into n shares
  - k < n shares are needed for decryption

    (reconstruct *master key*)
  - formal term: *(k,n) threshold scheme*



Adi Shamir et al.:
How to share a secret

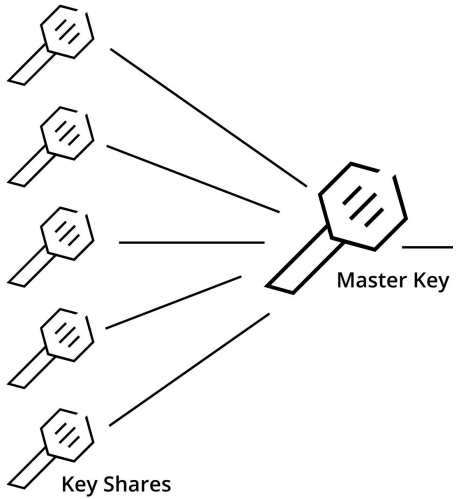
Picture

Shamir's Secret Sharing - A (k,n) threshold scheme

Given a secret ***S***, Goal: Divide ***S*** into ***n*** pieces $S_1,\ldots,S_n$ so that:

(1): knowledge of any ***k*** or more S_i pieces makes ***S*** easily computable

(2): knowledge of ***k-1*** or fewer S_i pieces leaves ***S*** completely undetermined



Adi Shamir et al.:
How to share a secret

Picture

A simple (k,n) Threshold Scheme

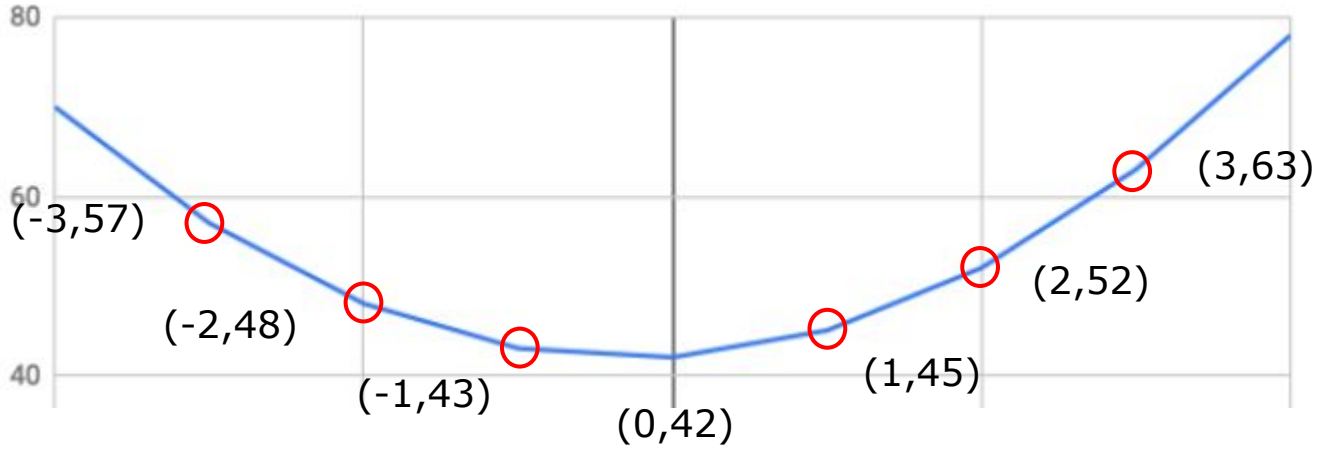
- given k (different) points in the 2-dimensional plane (x_i,y_i)
  ->there is one and only one polynomial q(x) of degree k-1 such that $q(x_i) = y_i$ for all i
- pick a random k-1 degree polynomial: $q(x) = a_0 + a_1x + \ldots + a_nx^{(k-1)}$
- set $a_0 = \mathbf{S} = q(0)$
- set $D_1 = q(1),\ldots,D_n = q(n)$
- Given any k subset of D_i values (together with indices) we can find q(x) and so q(0) which is our secret S

Adi Shamir et al.:
How to share a secret

Shamirs's Secret Sharing

A simple (k,n) Threshold Scheme - Example

- Given Secret S = 42, want to have n = 6 and k = 3
- we choose 2 random numbers: 1 and 2
- $q(x) = 42 + 1x + 2x^2$
- pick 6 random (x,q(x)) tuples

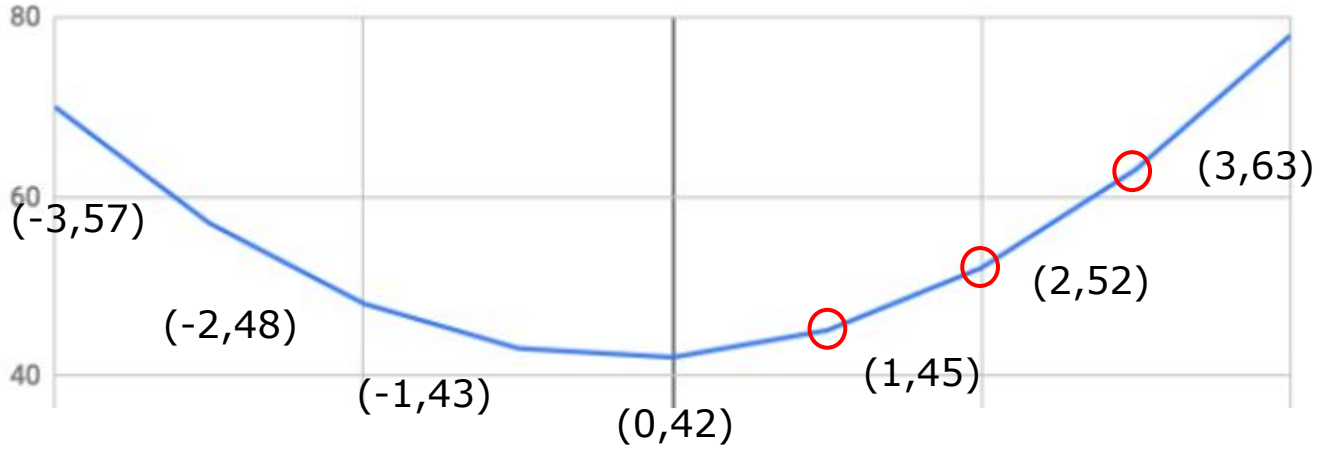


Adi Shamir et al.:
How to share a secret

Shamirs's Secret Sharing

A simple (k,n) Threshold Scheme - Example

- we can use any 3 out of the 6 points to reconstruct q(x)
  and in consequence reconstruct q(0) = 42 = S

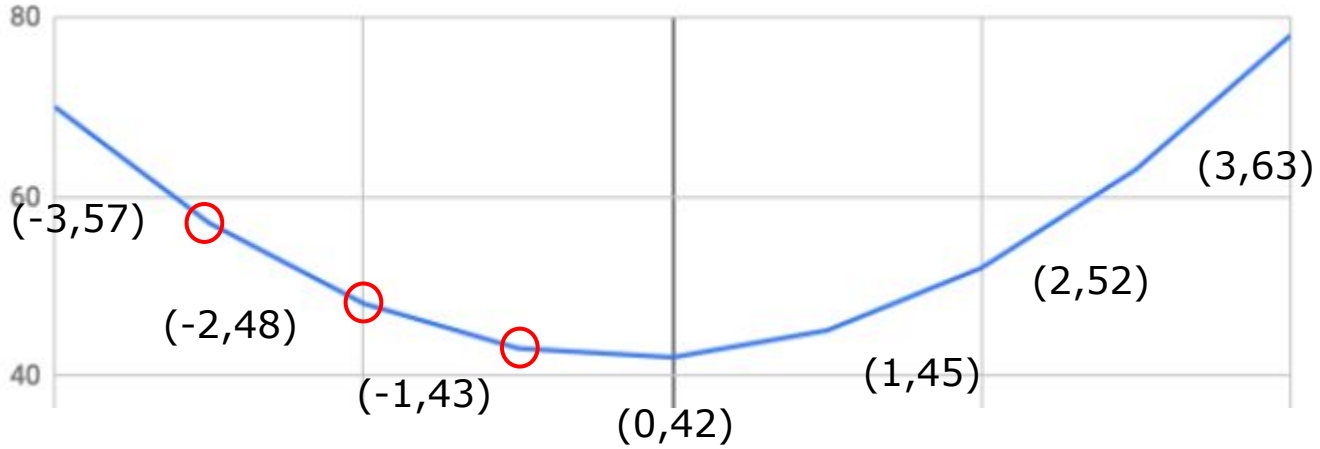


Adi Shamir et al.:
How to share a secret

Shamirs's Secret Sharing

A simple (k,n) Threshold Scheme - Example

- we can use any 3 out of the 6 points to reconstruct q(x)
  and in consequence reconstruct q(0) = 42 = S

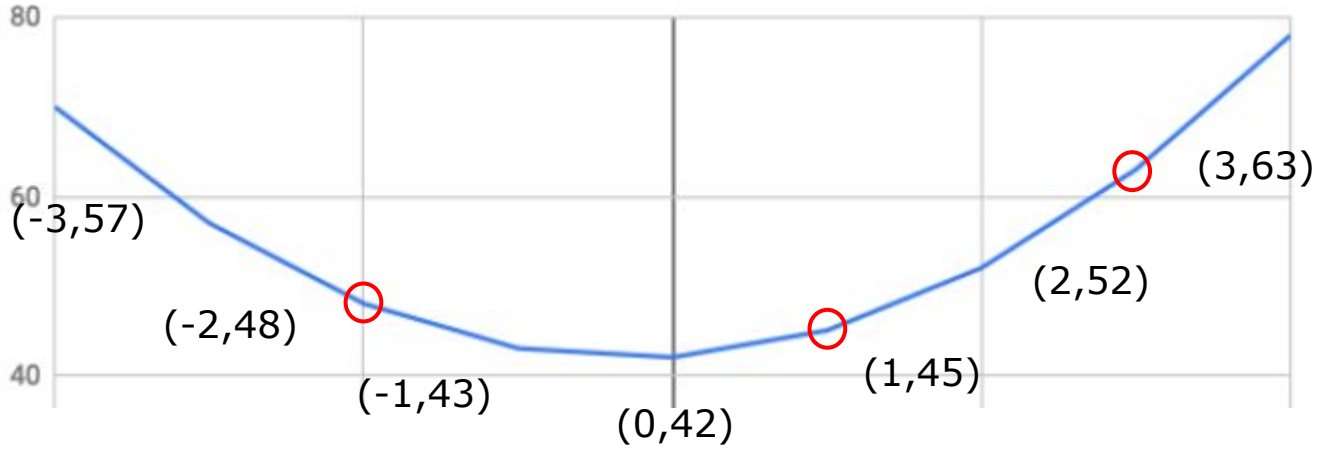


Adi Shamir et al.:
How to share a secret

Shamirs's Secret Sharing

A simple (k,n) Threshold Scheme - Example

- we can use any 3 out of the 6 points to reconstruct q(x)
  and in consequence reconstruct q(0) = 42 = S

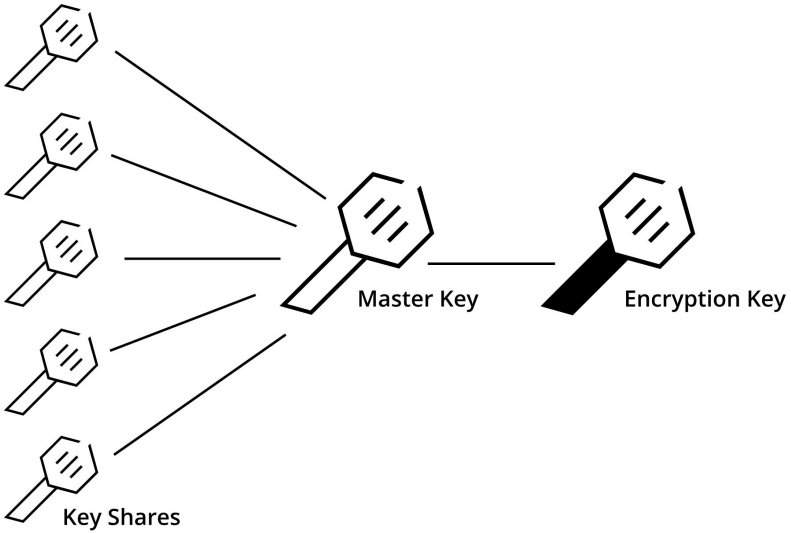


Adi Shamir et al.:
How to share a secret

Shamirs's Secret Sharing

Shamir’s Secret Sharing - Vault

- Vault uses *Encryption Key* to encrypt data in storage backend
- *Encryption Key* is de-/encrypted by *Master Key*
- *Master Key* is split into several *Key Shares*



Shamir’s Secret Sharing - Vault

Initializing the Vault

```
$ vault operator init

Unseal Key 1: 4jYbl2CBIv6SpkKj6Hos9iD32k5RfGkLzlosrrq/JgOm
Unseal Key 2: B05GlDRtfYckFV5BbdBvXq0wkK5HFqB9g2jcDmNfTQiS
Unseal Key 3: Arig0N9rN9ezkTRo7qTB7gsIZDaonOcc53EHo83F5chA
Unseal Key 4: 0cZE0C/gEk3YHaKjIWxhyyfs8REhqkRW/CSXTnmTilv+
Unseal Key 5: fYhZOseRgzxmJCmIqUdxEm9C3jB5Q27AowER9w4FC2Ck

Initial Root Token: s.KkNJYWF5g0pomcCLEmDdOVCW

Vault initialized with 5 key shares and a key threshold of 3. Please securely
distribute the key shares printed above. When the Vault is re-sealed,
restarted, or stopped, you must supply at least 3 of these keys to unseal it
before it can start servicing requests.

Vault does not store the generated master key. Without at least 3 key to
reconstruct the master key, Vault will remain permanently sealed!
```

Secret Sharing - Choosing values for k and n

- How could a single user access the master key
  - we should really think about this first
  - give k shares to an “admin” user
  - deactivate secret sharing
- Give one user more power than other users
  - e.g. k = 5, give him 3 key shares instead of one

    -> only has to contact 2 other users
- Given that each user has **only one** key share
  - system is protected if less than k users are malicious
  - we can use very high k (e.g. k = number of users)

A simple (k,n) Threshold Scheme - Problems

- Using integer arithmetic
  - makes it easy to make good guesses about secrets
  - e.g. n=6, k=3, 2 of 3 points are known to an attacker:

    only 150 possible values for the secret
  - what the definition says:
    (2): knowledge of ***k-1*** or fewer S_i pieces
    leaves ***S*** completely undetermined

- Solution: using finite field arithmetic
  - "endlicher Körper"
  - finite set of elements
  - defined: multiplication, addition, subtraction and division
  - satisfying field axioms

Shamir's Secret Sharing in practice - Vault

```go
// Split takes an arbitrarily long secret and generates a `parts`
// number of shares, `threshold` of which are required to reconstruct
// the secret. The parts and threshold must be at least 2, and less
// than 256. The returned shares are each one byte longer than the secret
// as they attach a tag used to reconstruct the secret.
func Split(secret []byte, parts, threshold int) ([][]byte, error) {
```

```go
// Combine is used to reverse a Split and reconstruct a secret
// once a `threshold` number of parts are available.
func Combine(parts [][]byte) ([]byte, error) {
```

Vault: shamir.go

```go
// interpolatePolynomial takes N sample points and returns
// the value at a given x using a lagrange interpolation.
func interpolatePolynomial(x_samples, y_samples []uint8, x uint8) uint8 {
```

```go
// makePolynomial constructs a random polynomial of the given
// degree but with the provided intercept value.
func makePolynomial(intercept, degree uint8) (polynomial, error) {
```

Do they use integer arithmetic ?!?

Shamir's Secret Sharing in practice - Vault

```
113  // mult multiplies two numbers in GF(2^8)
114  func mult(a, b uint8) (out uint8) {
```

```
83  // div divides two numbers in GF(2^8)
84  func div(a, b uint8) uint8 {
```

- GF(p^n) - finite field with p^n elements
- GF - Galois Field

Everything is okay

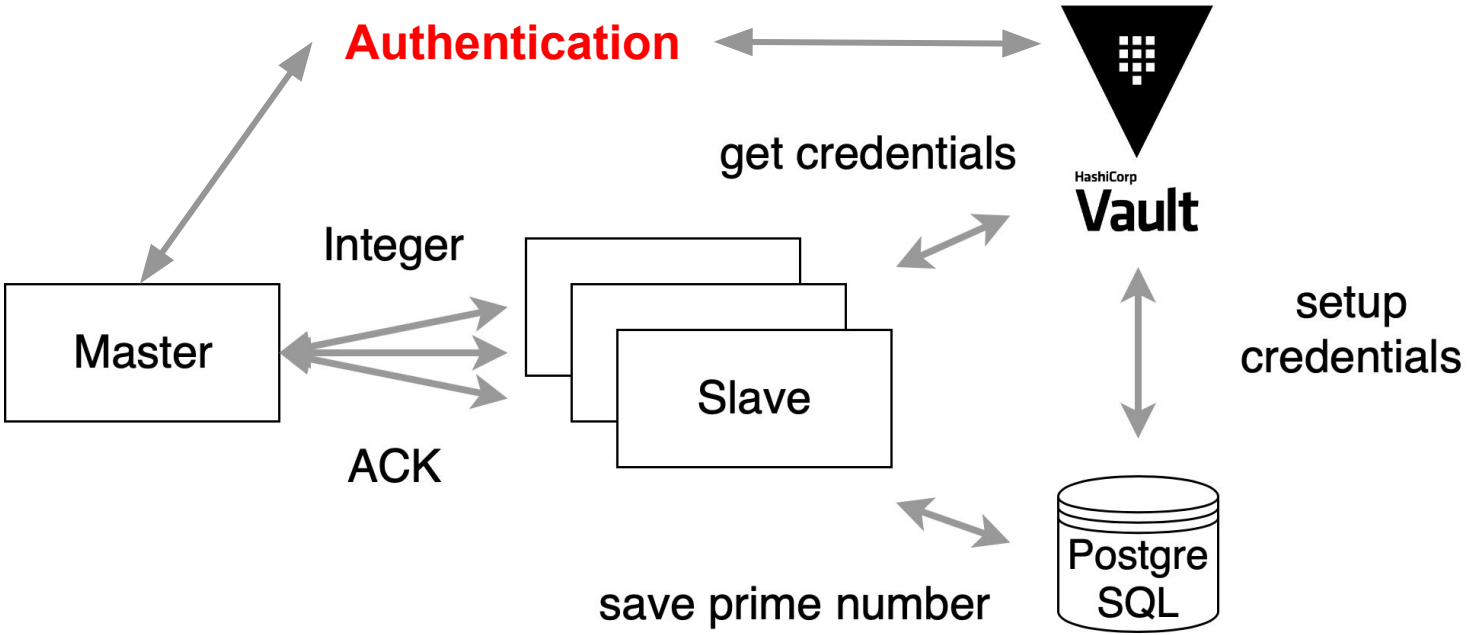
Évariste Galois - founder of finite field theory

[Vault: shamir.go]

[Finite field arithmetic]

[Evariste Galois]

Example - Architecture

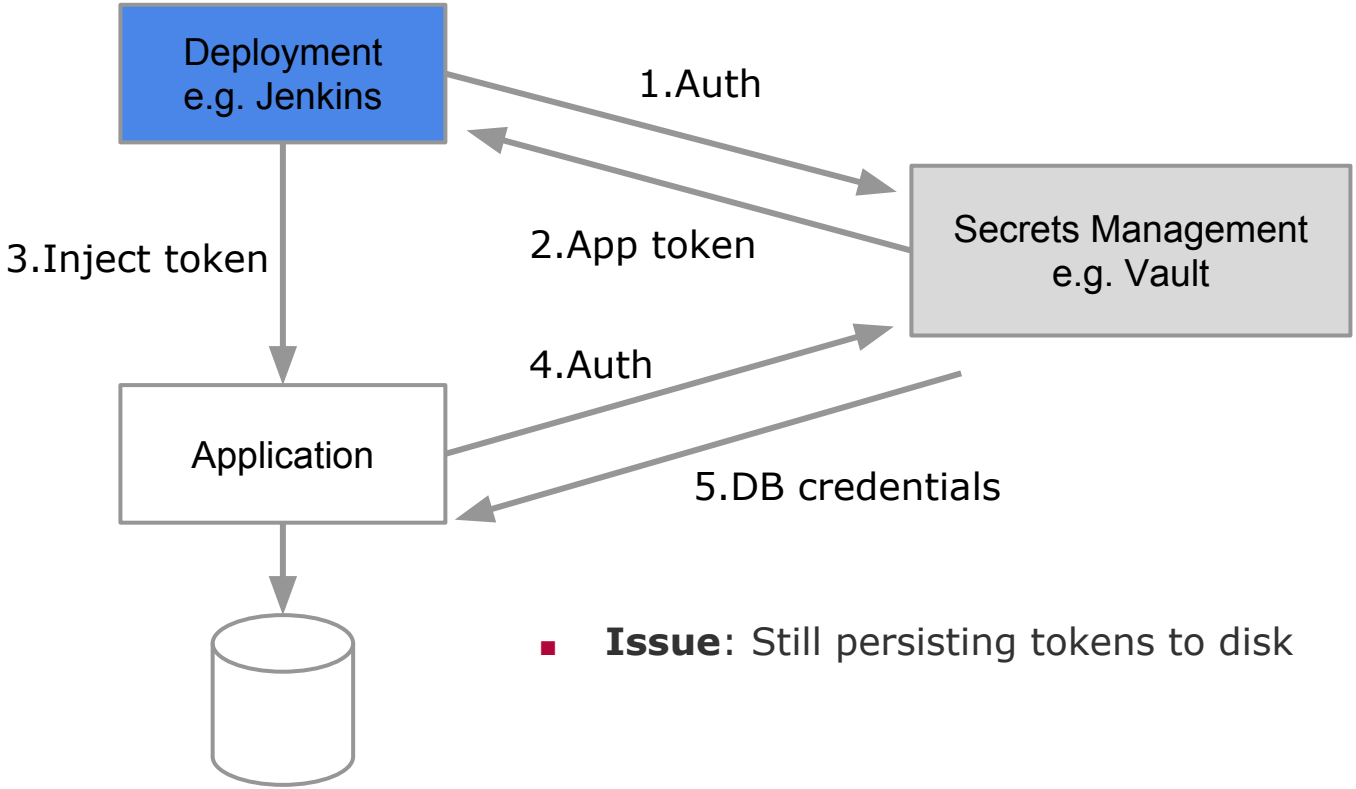


Challenges - Secret Zero

- For authentication between user (machine/person) a token is used
  - e.g. vault: "*vault token*"

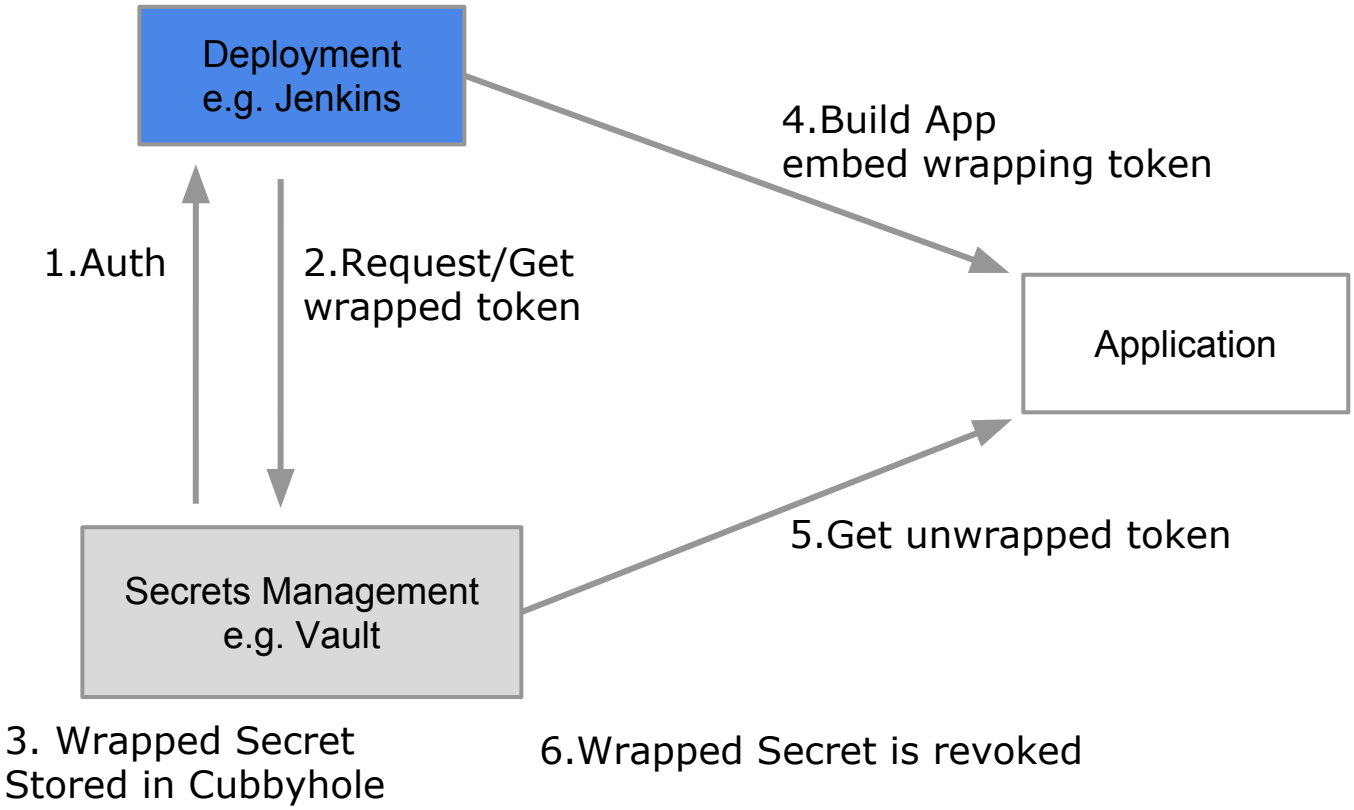
- But how does the system authenticates itself for the first time?
  - User needs some kind of *secret* for first authentication
  - We call this secret our "**Secret Zero**"
  - The problem of how to equip a system with the "**Secret Zero**"
    - Zero Knowledge Problem

Distribute Secrets - Naive Way



- **Issue**: Still persisting tokens to disk

Distribute Secrets - Wrapped Secrets



Distribute Secrets - Wrapped Secrets - Advantages

- Wrapping token only valid in a brief window between deployment/startup
- If the wrapping token is stolen
  - It’s no longer valid (after a successful introduction)
  - If the token is somehow intercepted, the request for the permanent token will fail!
    - Alert, warning of a potentially compromised secret

Distribute Secrets - Demo

Future Work

- How do other solutions differ?
  - AWS, Docker, Chef, ...
- Transferring ideas to different scenarios, e.g. end-2-end encryption
  - e.g. Gesundheitscloud
- Pentesting existing solutions
  - Analyse Thread Model

Achievements - Phase II

- Managing secrets is challenging!
  - Different approaches are available already
- Possibility of splitting and sharing secrets
  - Shamir Secret Sharing
- Practical implementation using Vault
  - Dynamic secret creation
  - Zero-Knowledge Problem
  - Cubbyhole / Wrapped Secrets

We dedicate this presentations to our mothers Marianne and Sabine whose maiden names shall not be relevant because this is a security presentation

Thank you
for your attention!

Johannes Kroschewski and Nils Straßenburg

Secrets Management

Network Security In Practice - Final Presentation