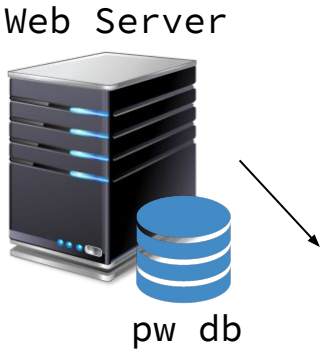
# SeifPass: a Secure Password Manager

**MIT PRIMES**
**By: Cristian Gutu**
**Computer Science**
**Mentor: Albert Kwon**

# Problem

- Passwords: common, but vulnerable to offline attacks


| User | Password |
|---|---|
| Alice | password123 |
| Bob | MyPetsName |
| ... | ... |

# Hash

Web Server

pw db

| User | Password |
|---|---|
| Alice | hash(password123) |
| Bob | hash(MyPetsName) |
| ... | ... |

Linked in

Attacker

pw db

| Common passwords | Computed hash |
|---|---|
| password123 | hash(password123) |
| MyPetsName | hash(MyPetsName) |
| ... | ... |

# Salted Hash

Web Server

pw db

| User | Hashed Password, Salt |
|---|---|
| Alice | hash(password123 + 4abcd@!#), 4abcd@!# |
| Bob | hash(MyPetsName + abcd%), abcd% |
| ... | ... |

Attacker

pw db

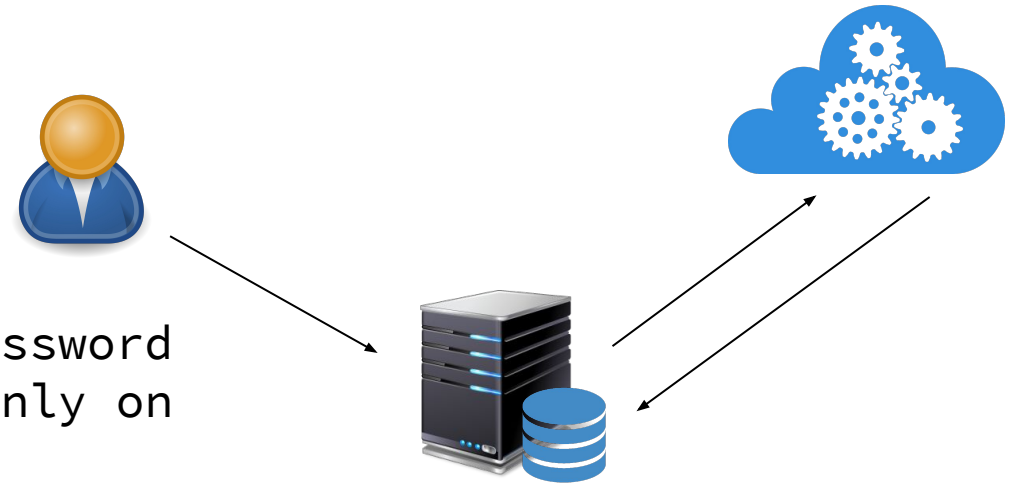
| Common passwords | Computed hash |
|---|---|
| password123 | hash(password123 + 4abcd@!#) |
| MyPetsName | hash(MyPetsName + abcd%) |
| ... | ... |

# Solution

- Architecture for password hardening using a remote cryptographic service

Benefits:

- No offline attacks



Hardening: encrypting password with secret key stored only on crypto service

# Distributed Trust


Hard to detect online attacks

harden($pw_0$)

harden($pw_1$)

User

hash(pw)

Web Server

pw db

hash(pw)

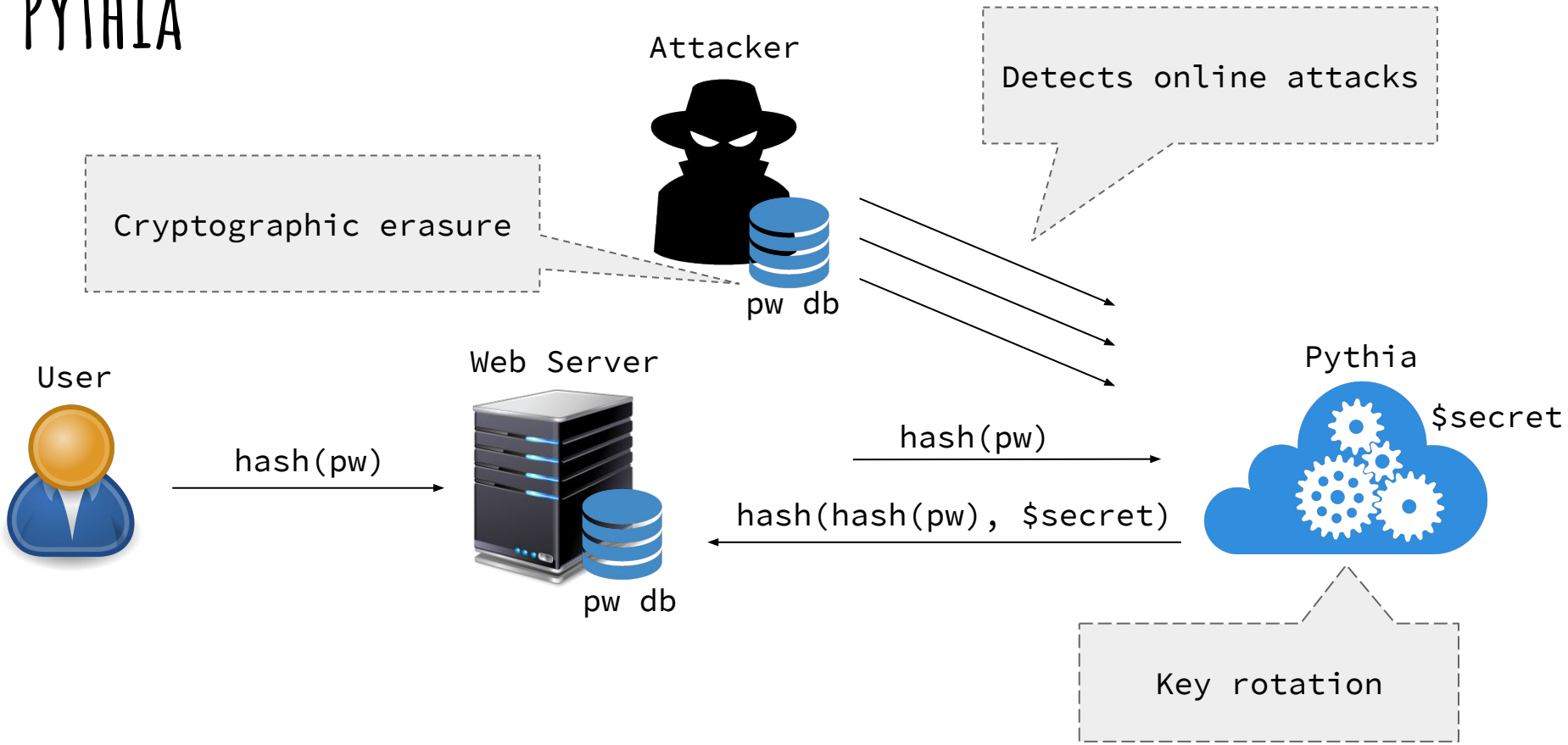
hash(hash(pw), $secret)

Crypto Service

$secret

How do we rotate $secret?

# Pythia

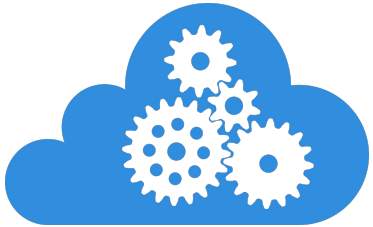

Attacker
Detects online attacks
Cryptographic erasure
pw db
User
Web Server
Pythia
$secret
hash(pw)
hash(pw)
hash(hash(pw), $secret)
pw db
Key rotation

# Pythia - New User

User

Web Server

pw db



Pythia

user,pw

t = random()
x = randomize(pw)

query(w, t, x)

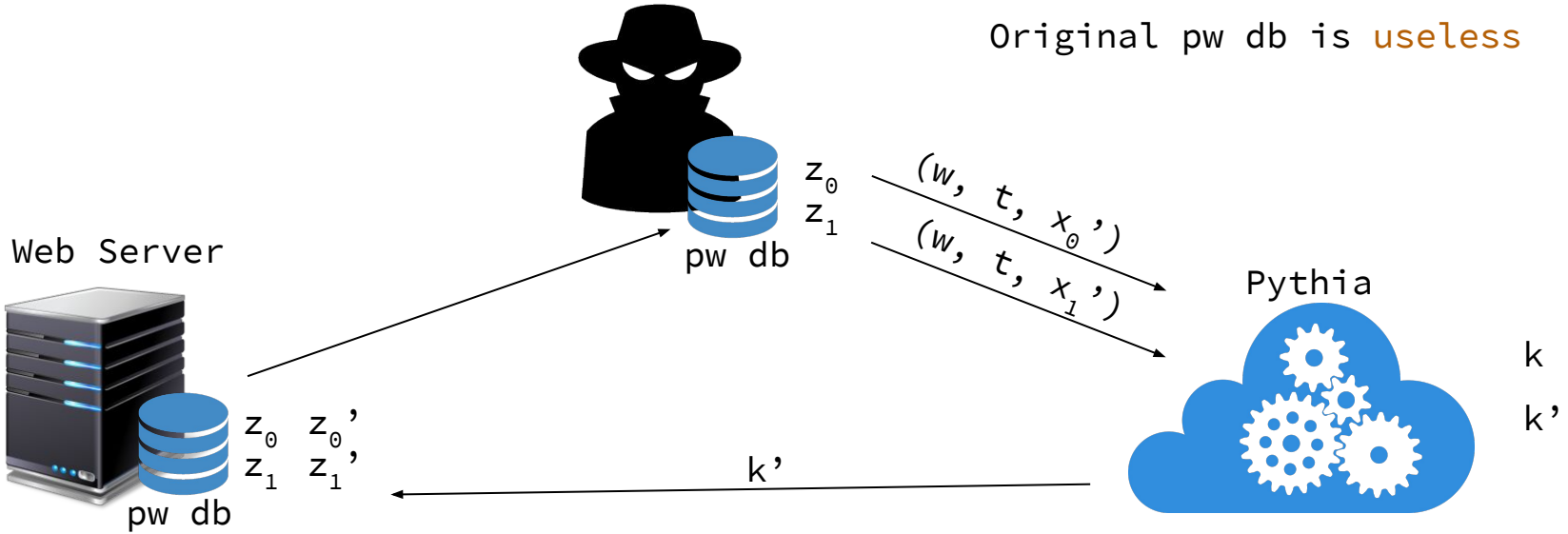
w: web server id
t: user id
x: randomized password

k = keytable[w]
y = F(k, t, x)

y

z = unrandomize(y)
store: (user, t, z)

# Compromise Recovery



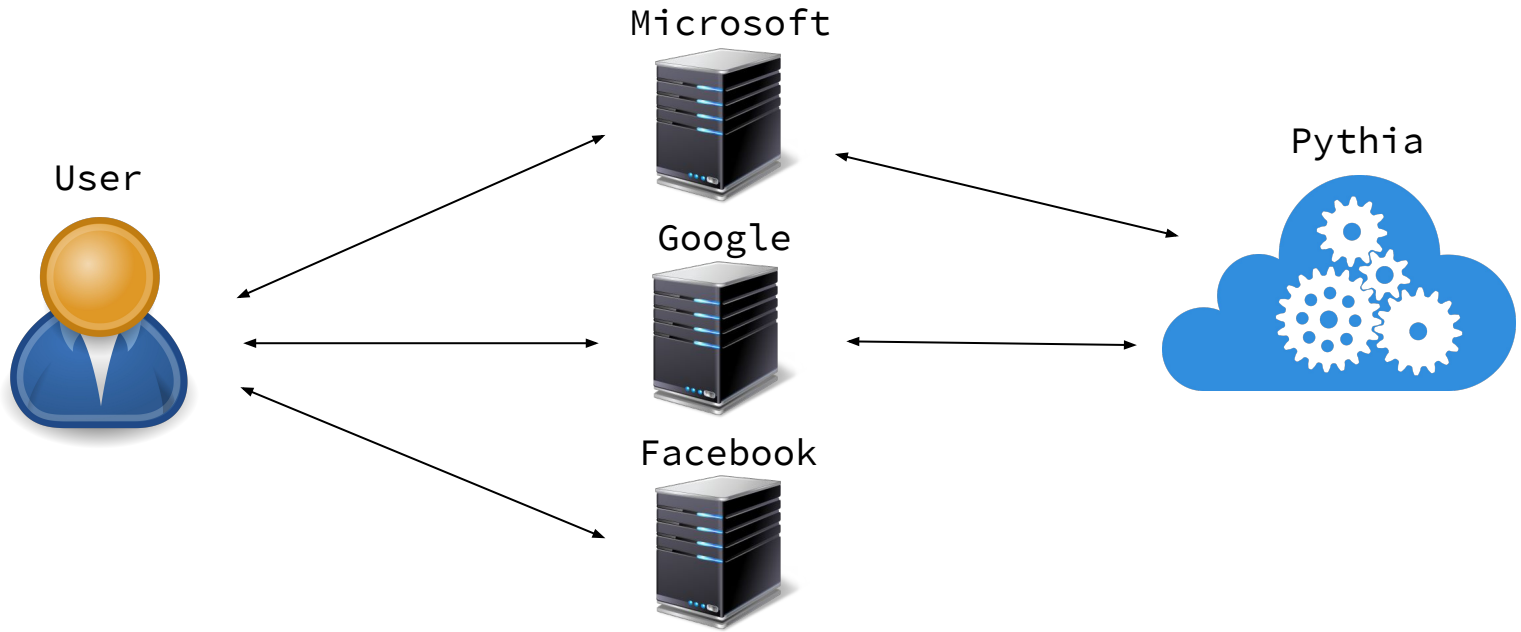

No need for original password

User password does not change

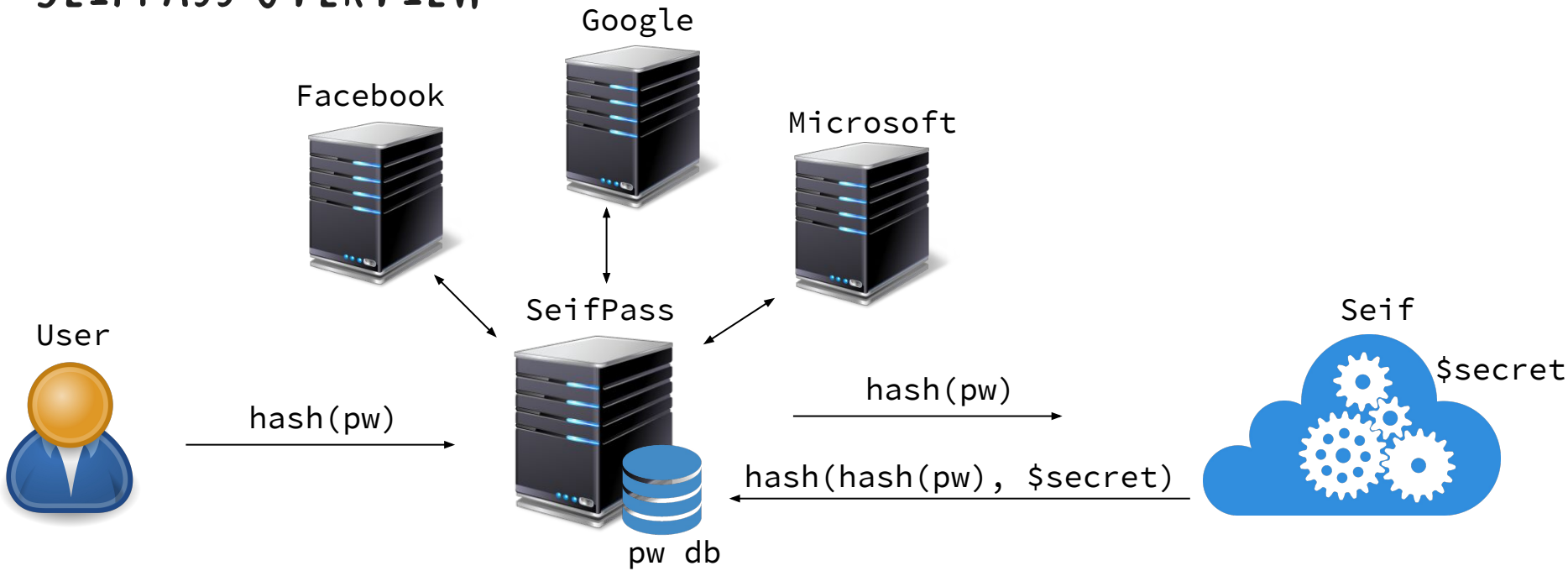
**Key:**
z_i : passwords
w: web server id
t: user id
x: randomized password
k: pythia instance for this web server

# Limitations of Pythia

- May be hard to merge in large code base
- No guarantee applications will implement it



# SeifPass Overview




User does not have to depend on application to secure password

# Seif - Node.js Pythia Wrapper

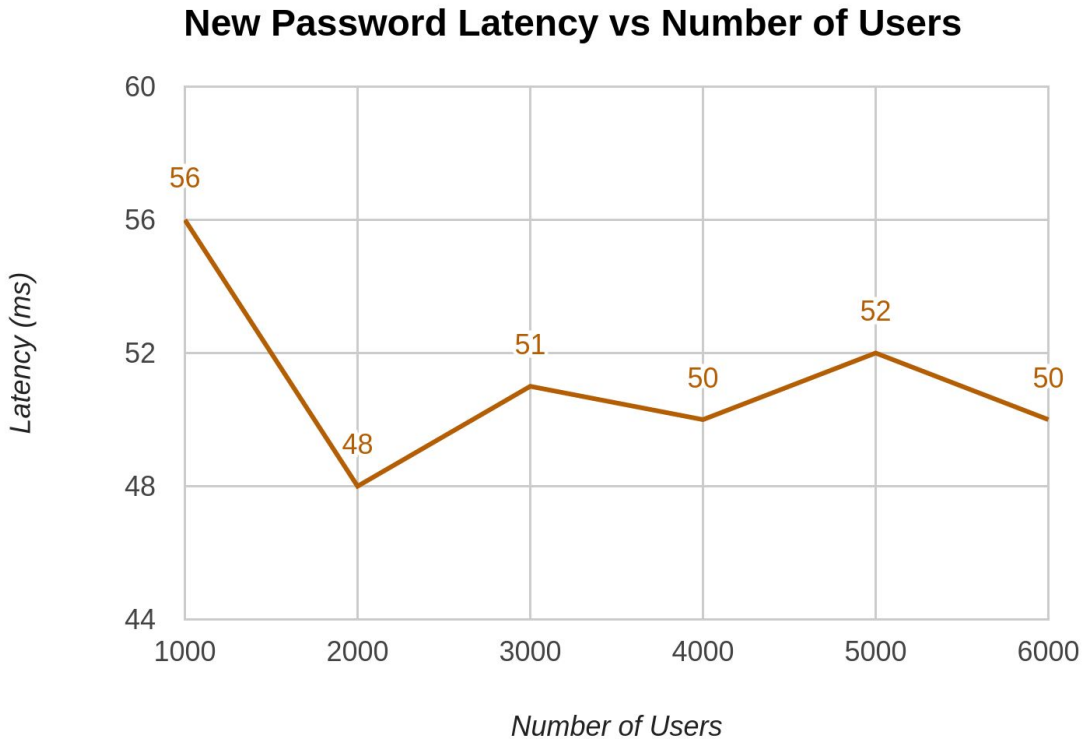
- Pythia Service in Node.js
- Easy to use
- https://github.com/naitsirc/seif

express

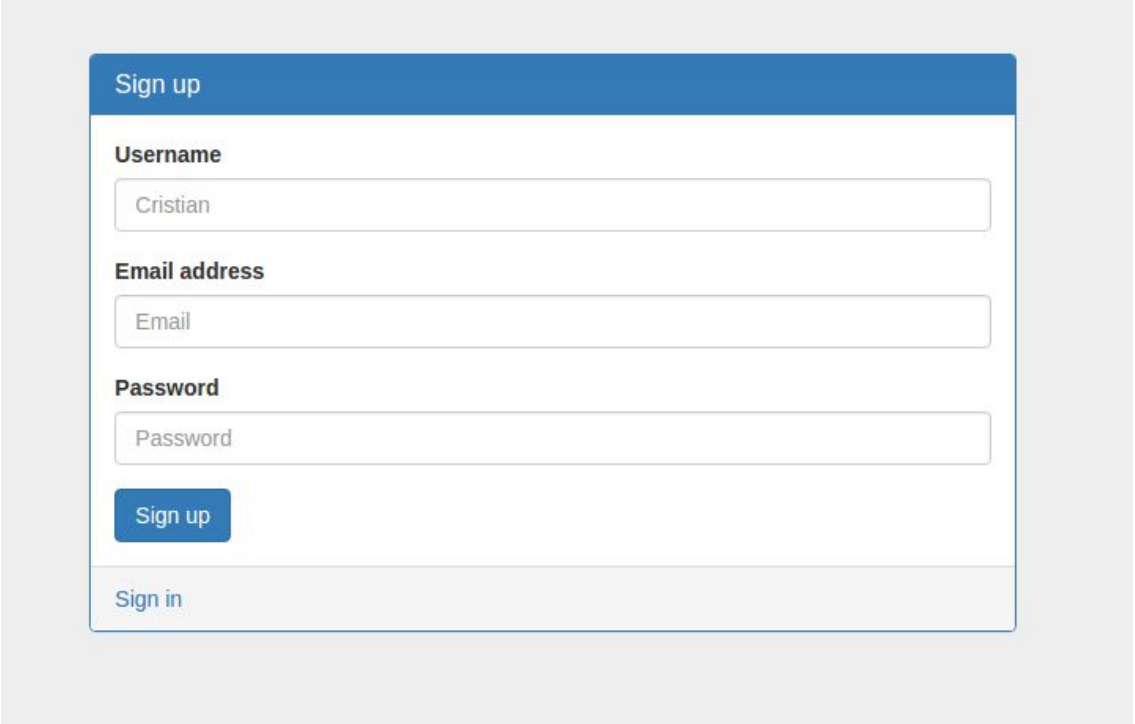
Dependencies:

- Relic Crypto Library
- pyrelic (python wrapper for Relic)
- Node.js
- Express.js (web application framework)
- mongoDB

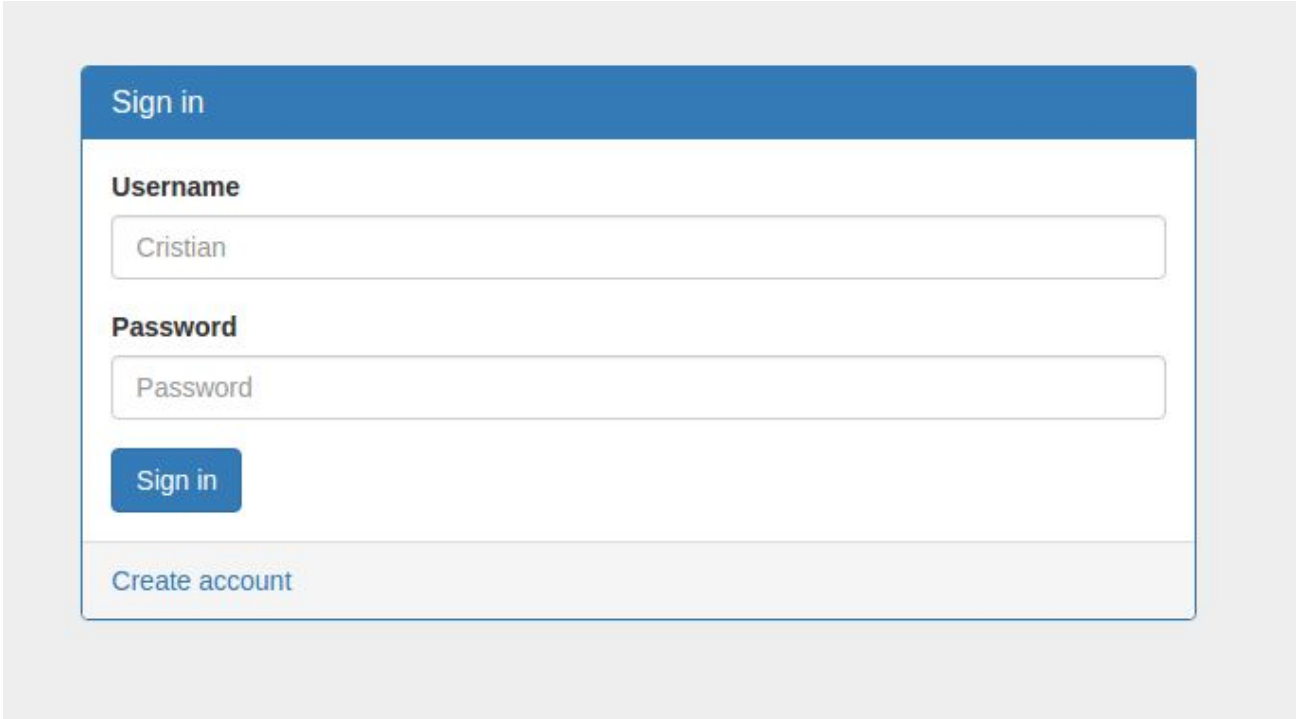
# Scalability Study



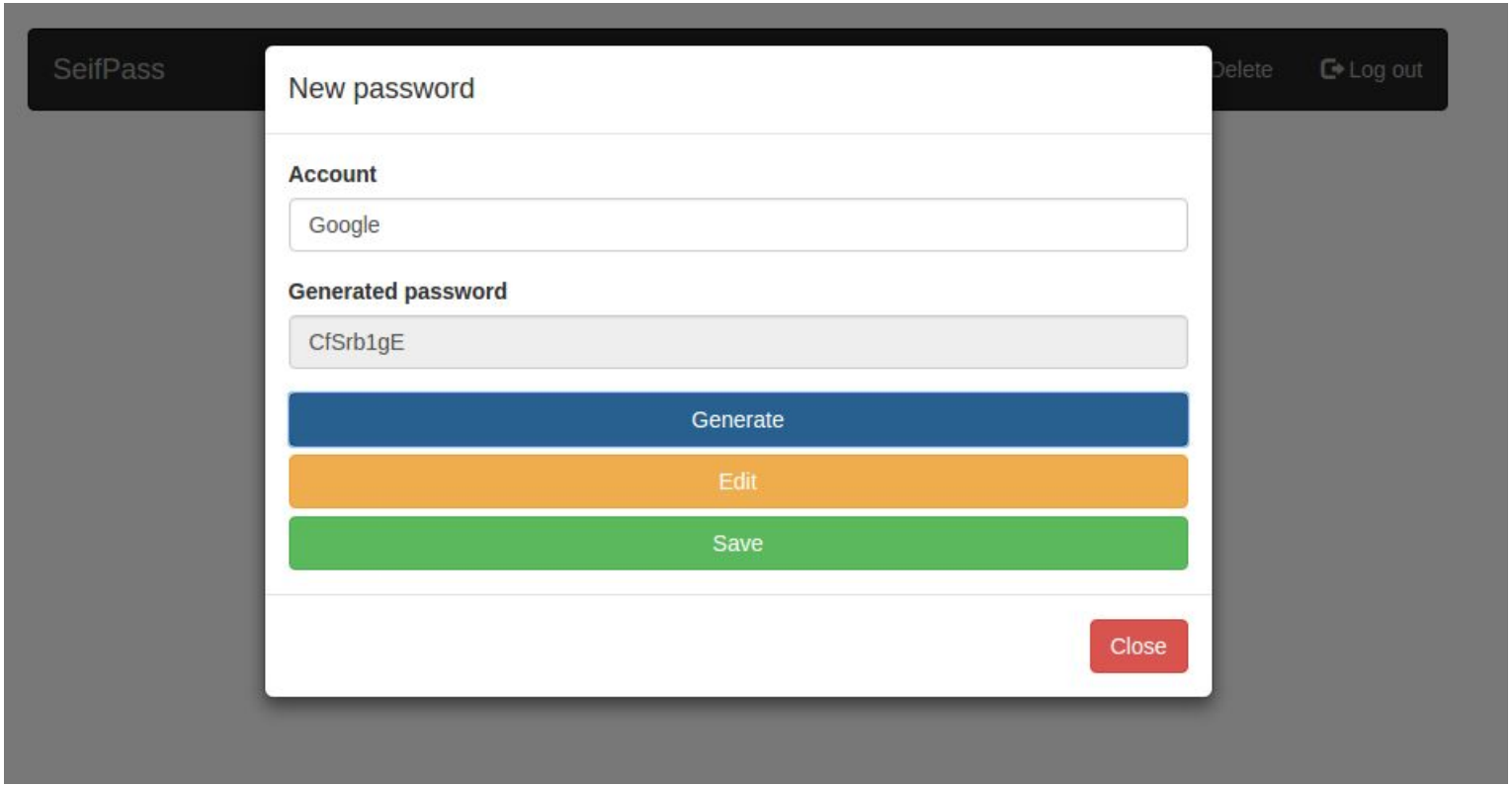
# SeifPass - Sign up



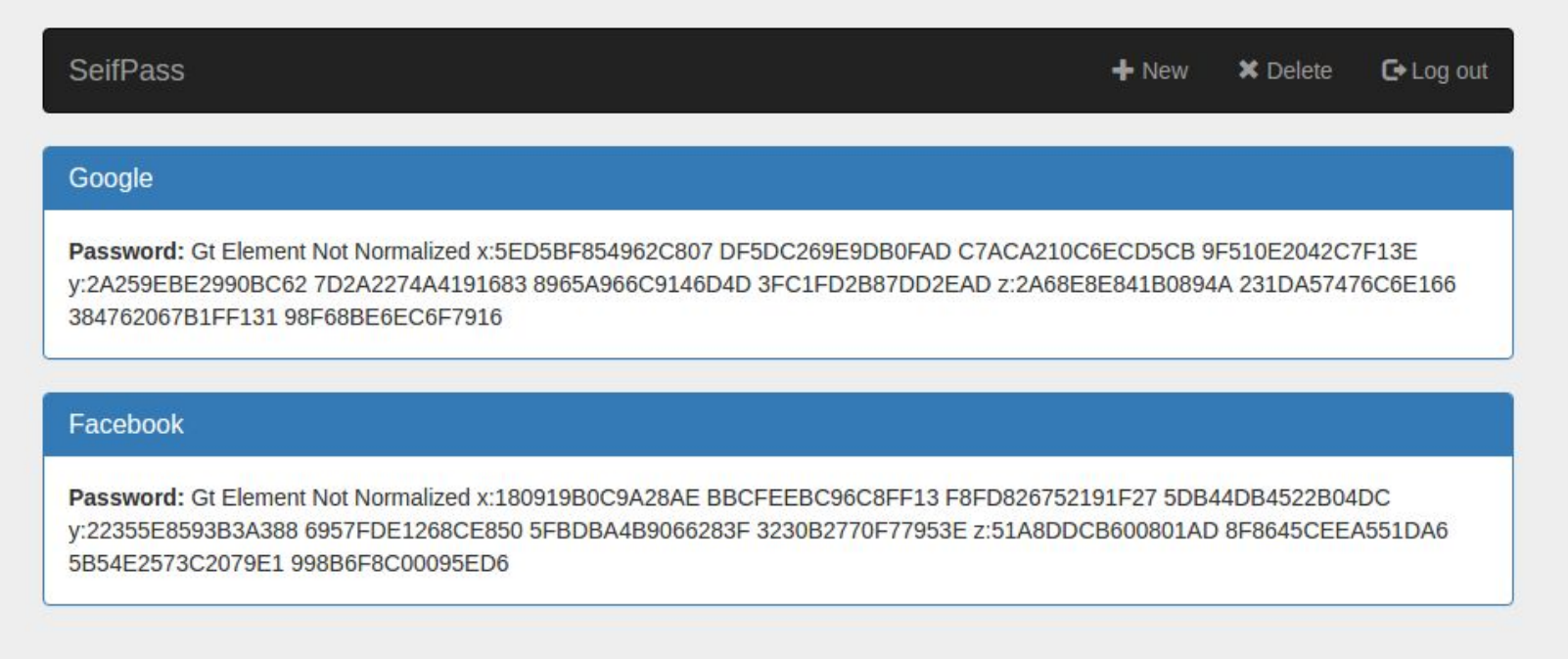
# SeifPass - Sign in



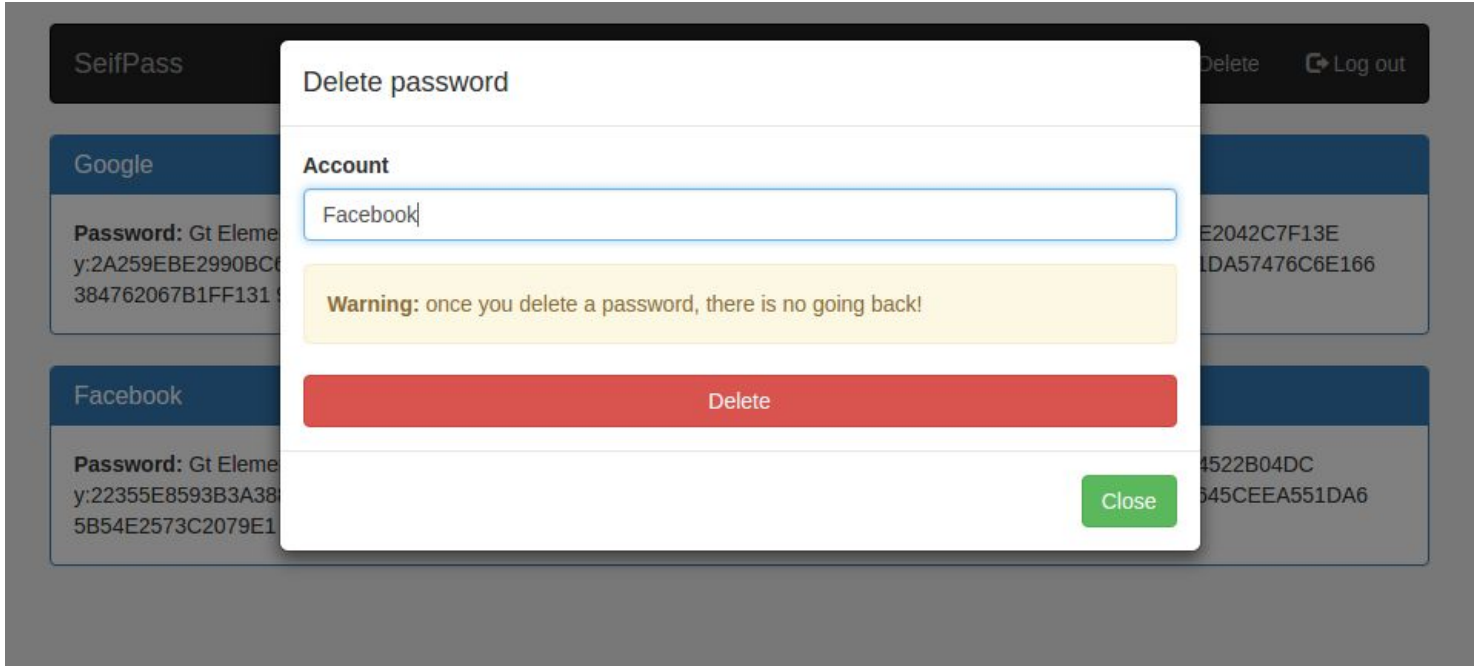
# SeifPass - New Password



# SeifPass - Password Manager



# SeifPass - Delete Password



# Acknowledgments

Thank you to Albert Kwon for mentoring and project suggestion!

Thank you Srini Devadas for the Computer Science Program!

Thank you MIT PRIMES for the opportunity!

Thank you to my parents for the support!