



# CRYPTOGRAPHY, the swiss army knife of secured APIs

#BzhCmpCrypto

Sébastien Lambour - @FinistSeb



# Too long title





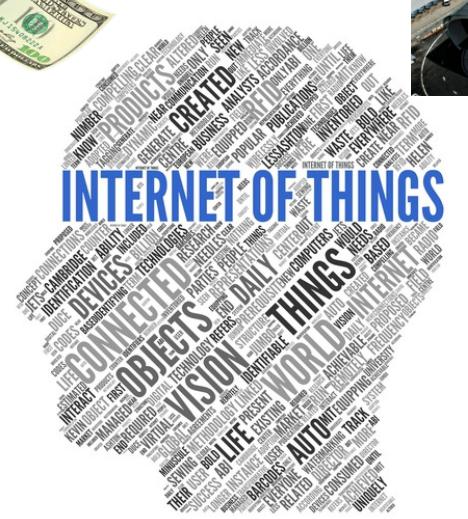
# Who am I





# Authorization & Authentication







Inspired from a true story



ONCE  
UPON A TIME

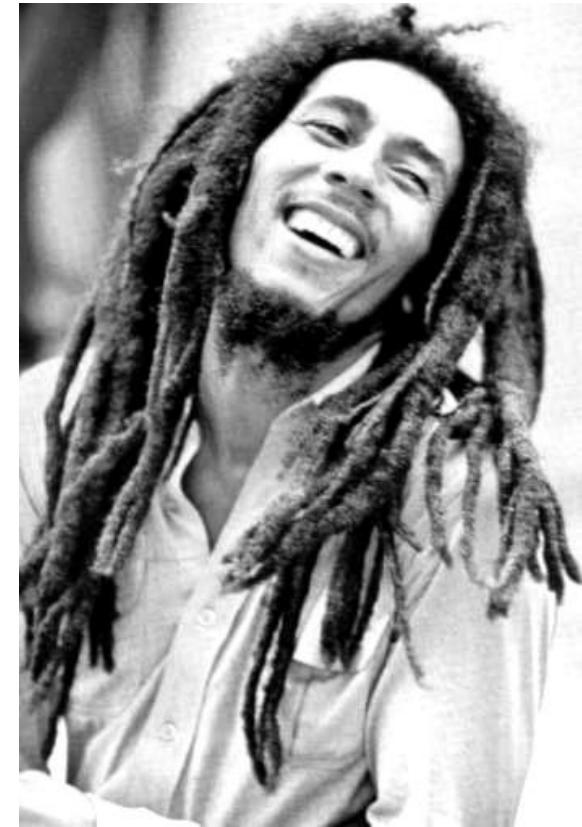
The logo for the television show "Once Upon a Time". It features the word "ONCE" in a large, stylized font where the "O" is a glowing blue circle and the "NCE" is in silver. Below it, the words "UPON A TIME" are written in a smaller, silver rectangular frame. The background is a dark, atmospheric forest scene with silhouettes of bare trees and a city skyline visible through the mist at the bottom.



**#LOYALTY**



# #TRANSFERT





# Banque JARAF (J'en ai rien à foutre)





# Alice to Bob

**FROM :** XXXFR-55555-1234567890-42

**TO :** XXXJA-12125-6662391327-69

**AMOUNT :** 100 €



# Are you sure?





# I'm serious... no?





# I loooooooooove HTTP post

POST / HTTP/1.1

content-type:application/x-www-form-urlencoded;charset=utf-8

host: https://raraf.bank

content-length:207

from=XXXFR-55555-1234567890-42

&to=XXXRU-22222-ANOTHER222-11

&amount=1000

&otp=713516

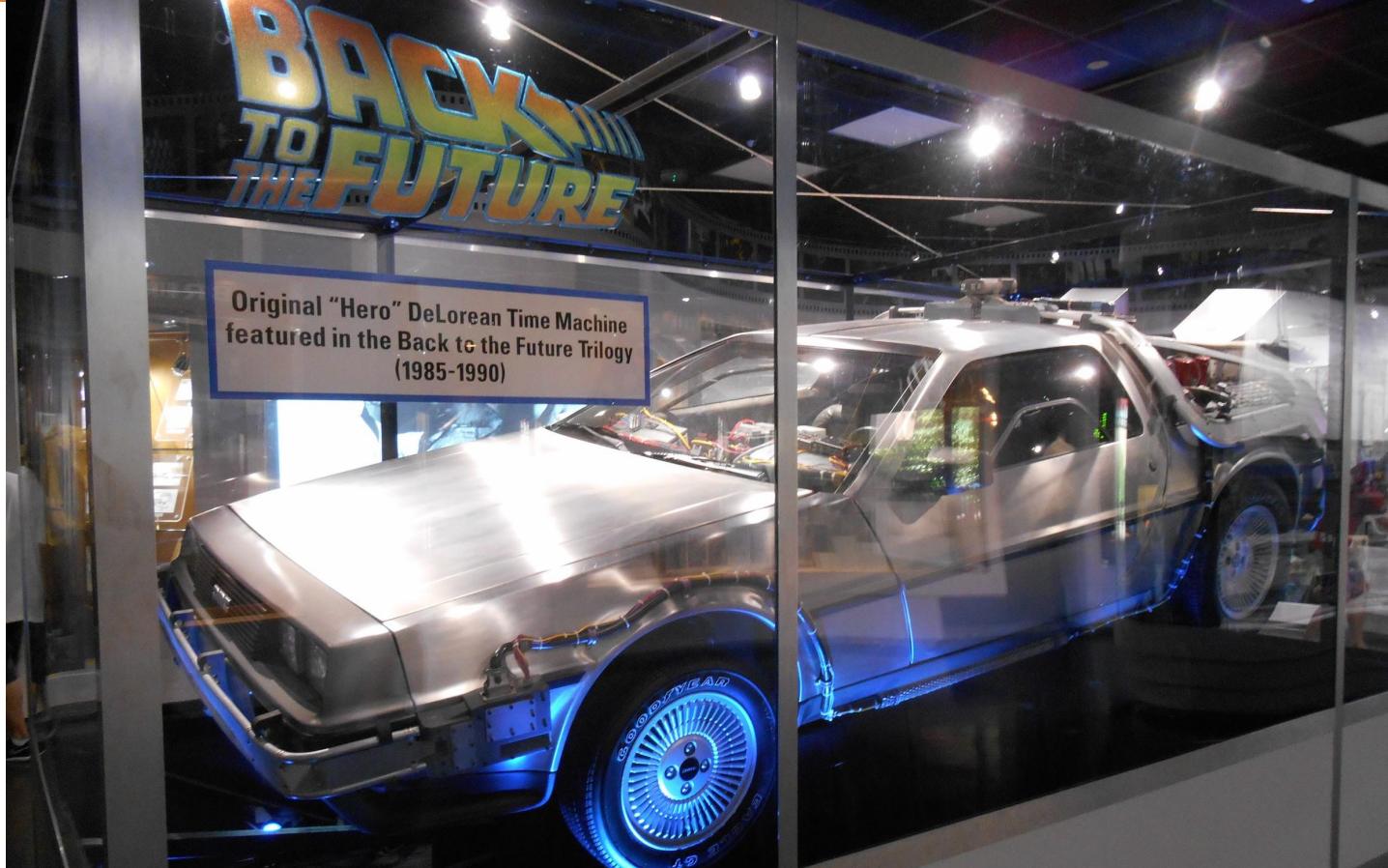


# WE DID IT!!!!





# Banque Back To The Future





# Alice to Bob

**FROM :** XXXFR-55555-1234567890-42

**TO :** XXXJA-12125-6662391327-69

**AMOUNT :** 100 €



# Seems not to bad...

**AUTHORISATION CODE : C6D8DF8E6A347473**



# Are you sure?





# I loooooooooove HTTP post

POST / HTTP/1.1

content-type:application/x-www-form-urlencoded;charset=utf-8

host: https://backtothefuture.bank

content-length:207

from=XXXFR-55555-1234567890-42

&to=XXXJA-12125-6662391327-69

&amount=100

&auth\_code=C6D8DF8E6A347800



# I'm serious... no ?





# I loooooooooove HTTP post

POST / HTTP/1.1

content-type:application/x-www-form-urlencoded; charset=utf-8

host: https://backtothefuture.bank

content-length:207

&otp=713516

&auth\_code=C6D8DF8E6A347800



# Seems serious... or not





# Coincidence ?

AUTHORISATION CODE : **C6D8DF8E6A347800**

AUTHORISATION CODE : **C6D8E76316324800**





# Too simple...

**28/5/2015 à 22:26:41**



**C6D8DF8CBABA14B3 -> 14328448010000405683**

**C6D8E76166B7E8C5 -> 14328456620000405701**



**28/5/2015 à 22:41:02**

0000405683  
0000405701



# SHALL WE PLAY A GAME?

GREETINGS PROFESSOR FALKEN.

Hello.

HOW ARE YOU FEELING TODAY?

I'm fine. You are you?

EXCELLENT. IT'S BEEN A LONG TIME. CAN YOU EXPLAIN  
THE REMOVAL OF YOUR USER ACCOUNT NUMBER ON 6/23/73?

People sometimes make mistakes

YES THEY DO. SHALL WE PLAY A GAME?

Love to. How about Global Thermonuclear War?

WON'T YOU PREFER A GOOD GAME OF CHESS?

■



# Play with timestamp



PAST

NOW-15

NOW

FUTURE





# Verbose API



PAST

NOW-15

NOW

FUTURE



{msg:'timestamp expired'}

{msg:'timestamp in the future'}



# Very verbose API



NOW-15

NOW

{msg:'sequence unknown'}

{msg:'sequence already validated'}

{msg:'bad user'}



# Play with timestamp





*Le thème c'est "Bounty Hunter",  
"chasseur de prime" en anglais, pas Bounty.*



LegoJeff



# #PROBLEM





# GRAILS

{ { <img alt="Green play button icon" data-bbox="158 718 281 938" />> } }



...



# #STATE

37%	XSS
16%	SQL Injection
5%	Path disclosure
4%	Denial of service attack
4%	Arbitrary code execution
4%	Memory corruption
3%	Cross-site request forgery
3%	Data breach
4%	File inclusion (Arbitrary, Local, Remote)
1%	Buffer overflow
15%	Other, including code injection (PHP/JS), etc



#1



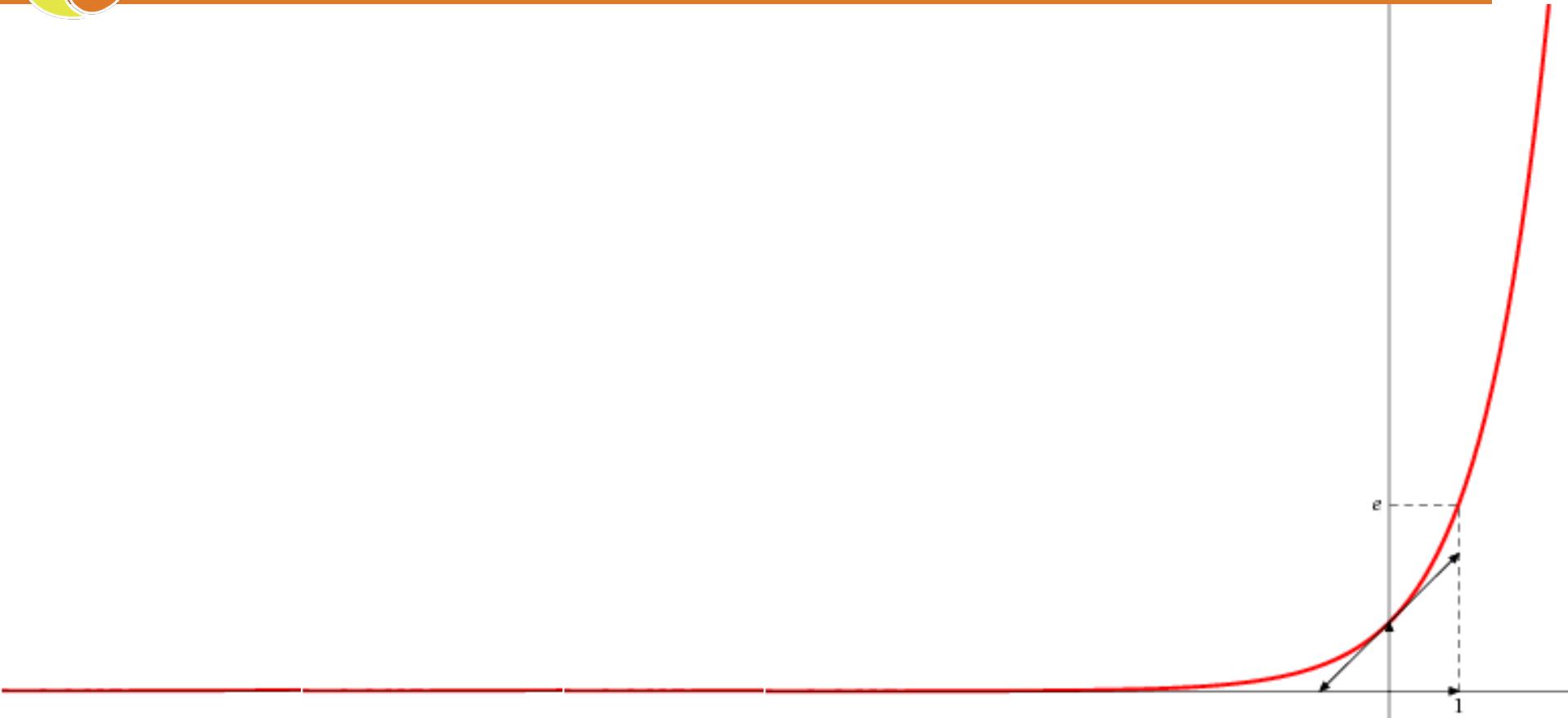


## #2 In the code (business)

```
    fcomplex<float> Conj(fcomplex<float> z)
    {
        fcomplex<float> RConj(fcomplex<float> z)
        {
            float w;
            if ((z.r == 0.0) && (z.i == 1.0))
                return 0.0;
            else
                c.i=0.0;
        }
        fcomplex<float> Cinv(fcomplex<float> z)
        {
            fcomplex<float> ans;
            if ((z.r == 0.0) && (z.i == 1.0))
                ans.r=1.0;
            else
                ans.r=z.r/z.r;
                ans.i=z.i/z.r;
            return ans;
        }
    }
}
```

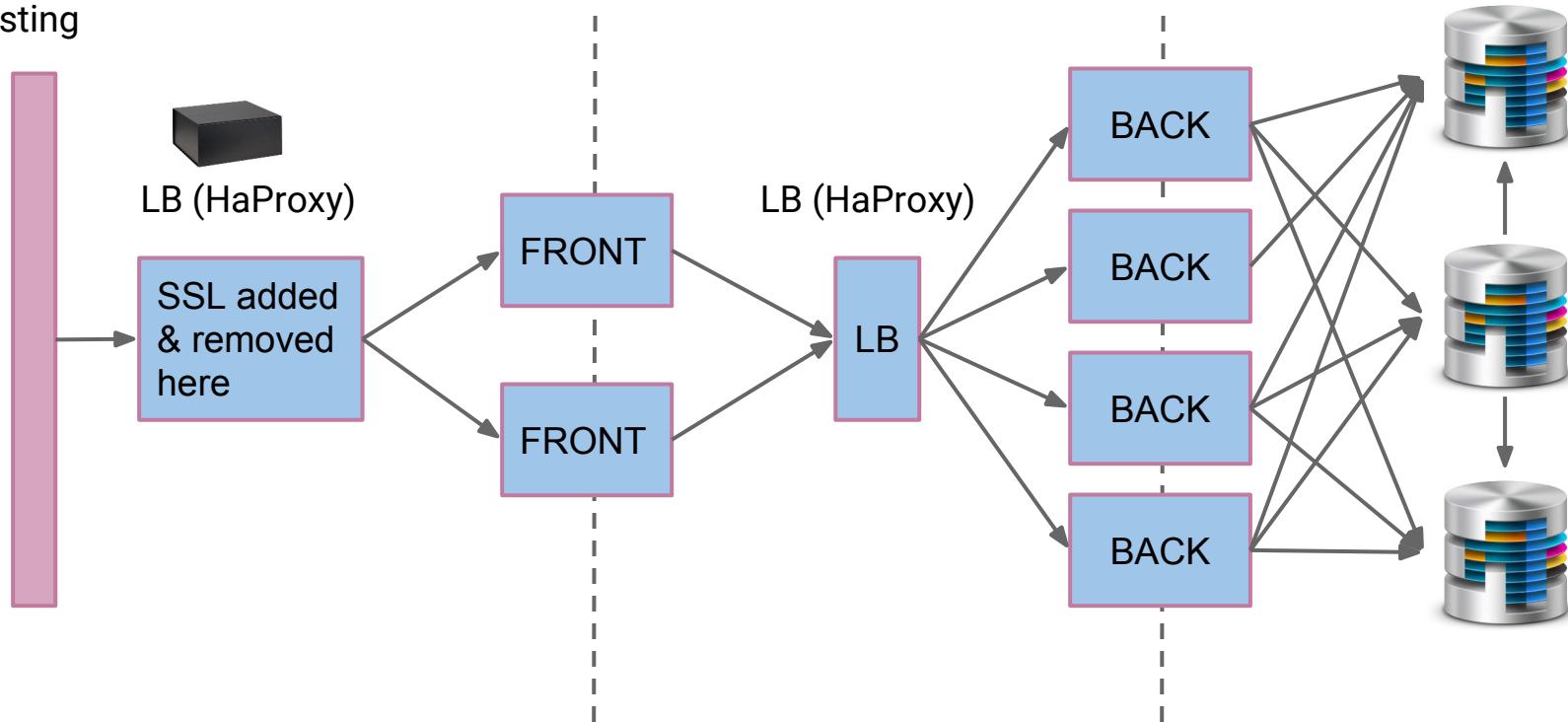


# #3 SCALABILITY





Hosting





Hosting

LB (HaProxy)

SSL added  
& removed  
here

FRONT

FRONT

LB (HaProxy)

LB

BACK

BACK

BACK

BACK

SHARD #1



SHARD #2



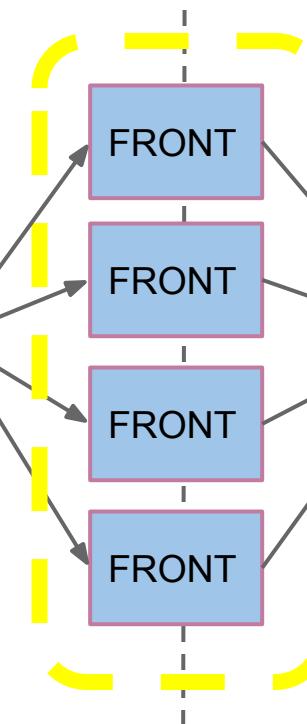
A vertical dashed line separates the front-end load balancers from the back-end servers.

A horizontal dashed line separates the two shards.



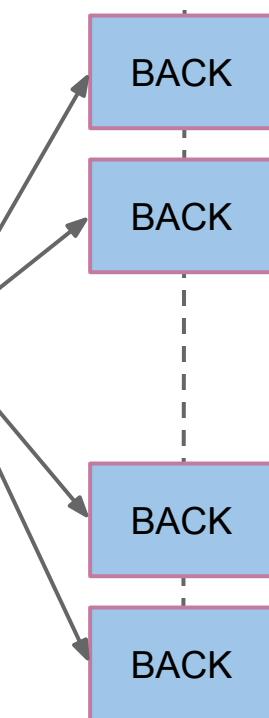
Hosting

LB (HaProxy)  
SSL added & removed here



LB (HaProxy)

LB



SHARD #1



SHARD #2



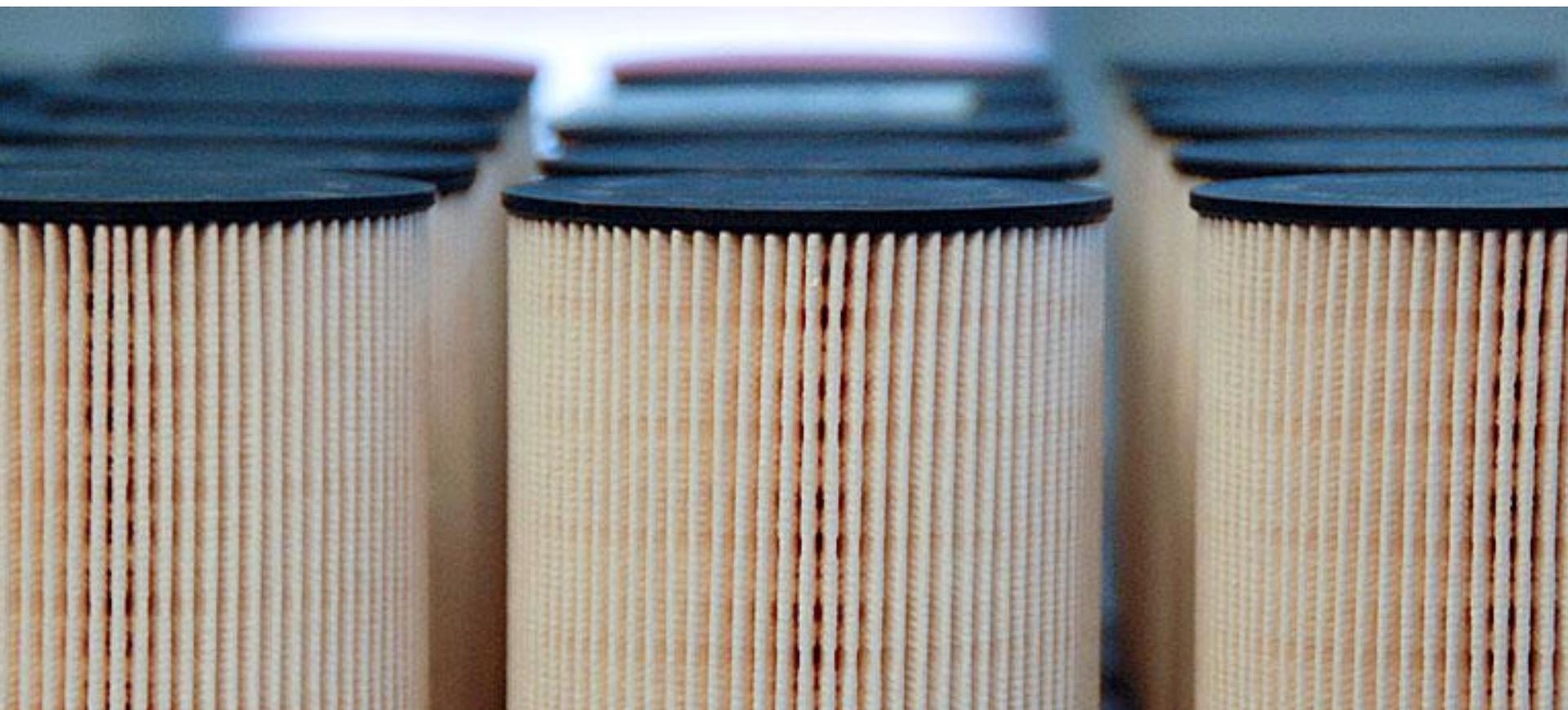


# #Crypto can help you

$$\int f(x)dx = \left( \sum_{j=1}^n a_j u_j(x) \right)' = \sum_{j=1}^n a_j u'_j(x) \quad \text{for } c=1, m \neq f(x), d=1, n \quad \int f(x)dx = \left( \sum_{j=1}^n a_j u_j(x) \right)' = \sum_{j=1}^n a_j u'_j(x) \quad \text{for } c=1, m \neq f(x), d=1, n$$
$$F = F(x_0 + \alpha x_0) - F(x_0) \quad I_1 = \int_{x_0}^{x_0 + \alpha x_0} \frac{dx}{x} \quad x \rightarrow a \quad x \rightarrow b \quad F = F(x_0 + \alpha x_0) - F(x_0) \quad I_1 = \int_{x_0}^{x_0 + \alpha x_0} \frac{dx}{x} \quad x \rightarrow a \quad x \rightarrow b$$
$$x_0 \pm y_0 \quad \left\{ \begin{array}{l} (V_{n+2})^{1-n} - (V_h)^{1-n} \\ \hline (V_h)^n \end{array} \right\} \sum_{k=0}^n a_k x^k \quad \left\{ \begin{array}{l} (V_{n+2})^{1-n} - (V_h)^{1-n} \\ \hline (V_h)^n \end{array} \right\} \sum_{k=0}^n a_k x^k \quad \left\{ \begin{array}{l} (V_{n+2})^{1-n} - (V_h)^{1-n} \\ \hline (V_h)^n \end{array} \right\} \sum_{k=0}^n a_k x^k \quad \left\{ \begin{array}{l} (V_{n+2})^{1-n} - (V_h)^{1-n} \\ \hline (V_h)^n \end{array} \right\} \sum_{k=0}^n a_k x^k$$
$$\left( 1 + \frac{1}{10^n + 1} \right)^{10^n + 1} < \left( 1 + \frac{1}{n} \right)^{n+1} \quad \psi\left(\frac{1}{10}\right) = [\psi\left(\frac{1}{10}\right)]^0 \quad \left( 1 + \frac{1}{10^n + 1} \right)^{10^n + 1} < \left( 1 + \frac{1}{n} \right)^{n+1} \quad \psi\left(\frac{1}{10}\right) = [\psi\left(\frac{1}{10}\right)]^0$$
$$= \int \pi f'(x)dx = \int \pi \left( \frac{f''(x)}{h^2} \right) dx + \int \frac{\pi f''(x)}{h^2} dx \cdot \Delta x \int [u_1(x) \cdot u_0(x) + \dots + u_n(x)] dx = \int \pi f'(x)dx = \int \pi \left( \frac{f''(x)}{h^2} \right) dx + \int \frac{\pi f''(x)}{h^2} dx \cdot \Delta x \int [u_1(x) \cdot u_0(x) + \dots + u_n(x)] dx$$
$$\sum_{k=0}^n a_k x^k = \sum_{k=0}^n a_k x^k \quad \left[ \frac{1}{3} + \frac{2}{4} + \frac{5}{7} + \frac{1}{11} \right] \quad P_h(z_0) = \sum_{k=0}^n a_k z_0^k \quad \left[ \frac{1}{3} + \frac{2}{4} + \frac{5}{7} + \frac{1}{11} \right] \quad P_h(z_0) = \sum_{k=0}^n a_k z_0^k \quad \left[ \frac{1}{3} + \frac{2}{4} + \frac{5}{7} + \frac{1}{11} \right]$$
$$\int f(x)dx \cdot C \cdot (x+x)^\alpha = \sum_{k=0}^n C_n^k x^{\alpha-k} x^k \int \left( \sum_{j=1}^n a_j f_j(x) \right) dx \quad \int f(x)dx \cdot C \cdot (x+x)^\alpha = \sum_{k=0}^n C_n^k x^{\alpha-k} x^k \int \left( \sum_{j=1}^n a_j f_j(x) \right) dx \quad \sum_{j=1}^n$$



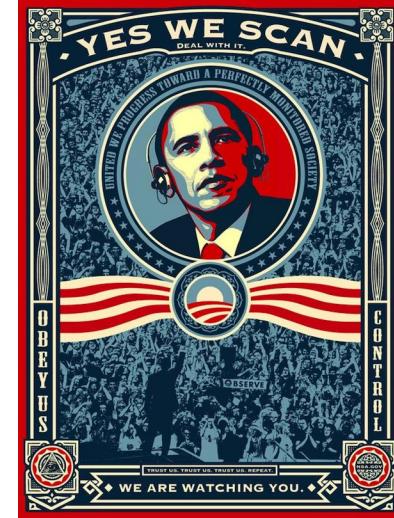
# #Crypto can help you







# Symmetric cryptography







# Cryptographic hash functions





# NEVER forget it





# With Salting





# Min salt size

**128 bit**



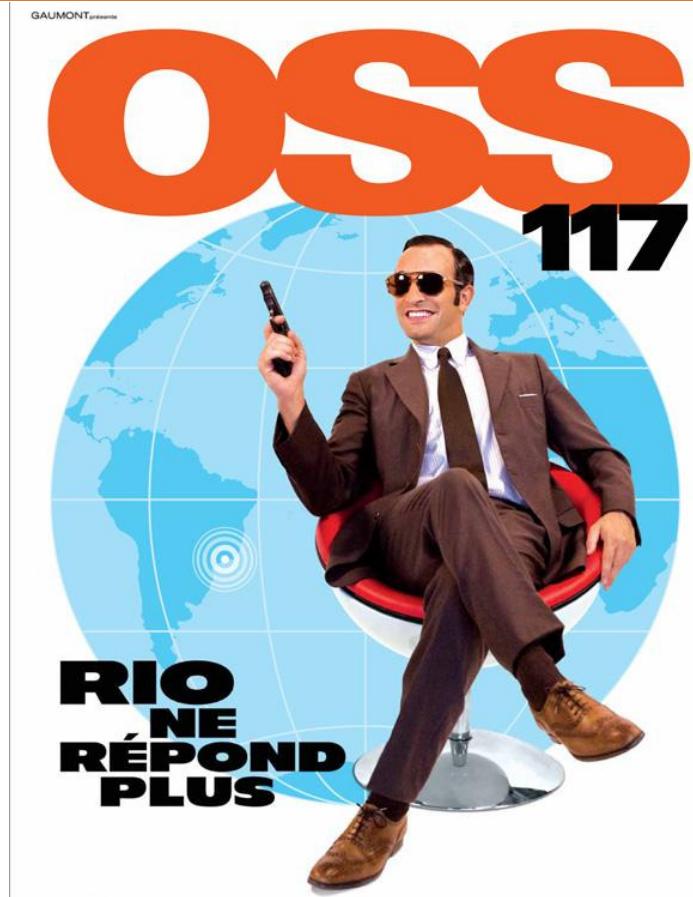


# Protect your keys





# Open Secret Server





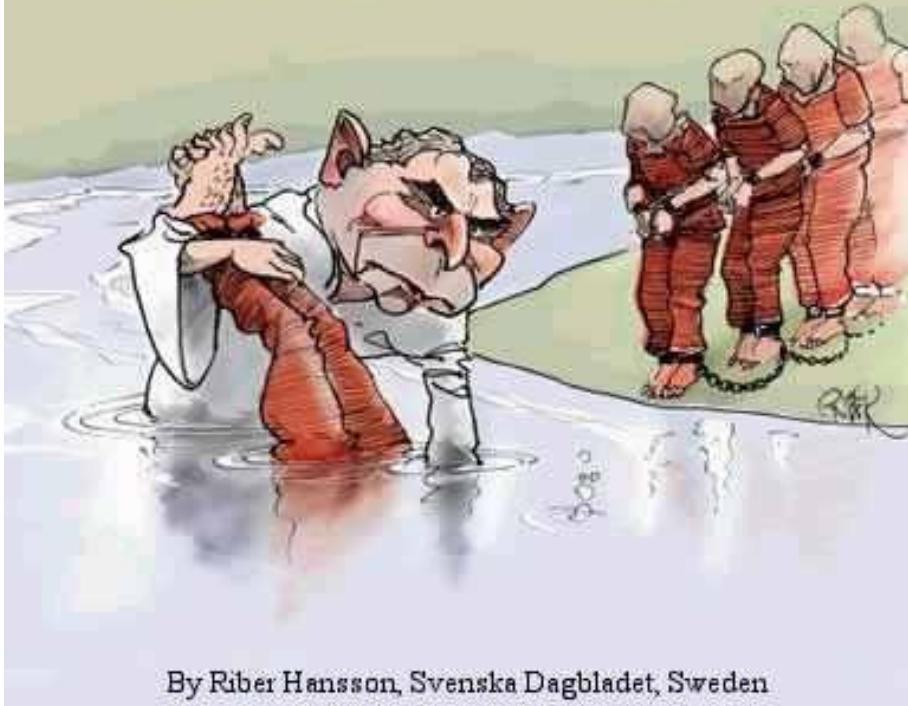
# Shamir algorithm





# #The limit...

**BUSH, "WE DON'T TORTURE"  
WE BAPTIZE!**



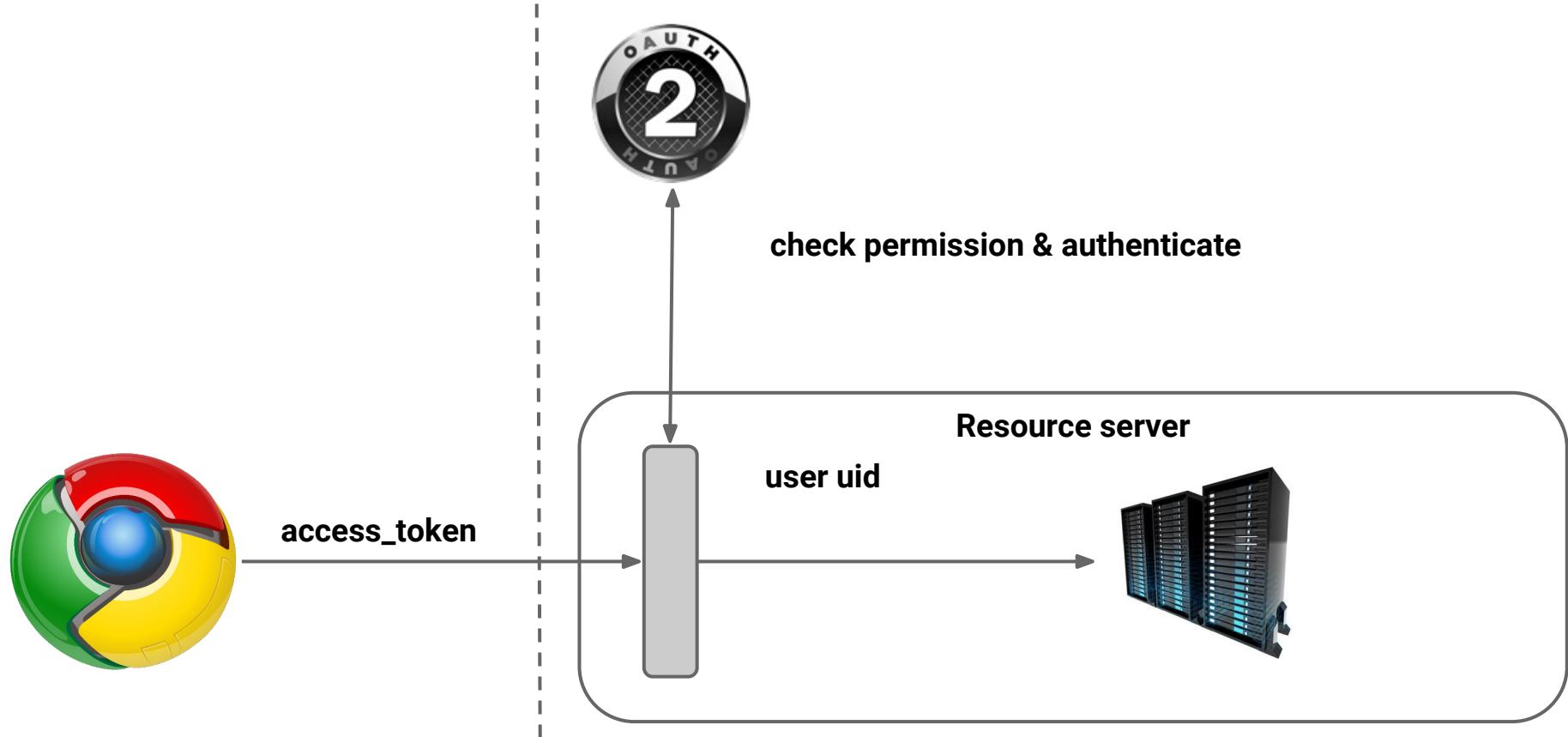
A photograph of three Adelie penguins standing on a white, snow-covered slope. They are black on top and white on the bottom, with a white patch on their wings. The penguin on the right is facing right, while the others are facing left. The background shows a bright blue sky with some white clouds.

# Go Ahead

# It Will Be Fun



# HTTP Request



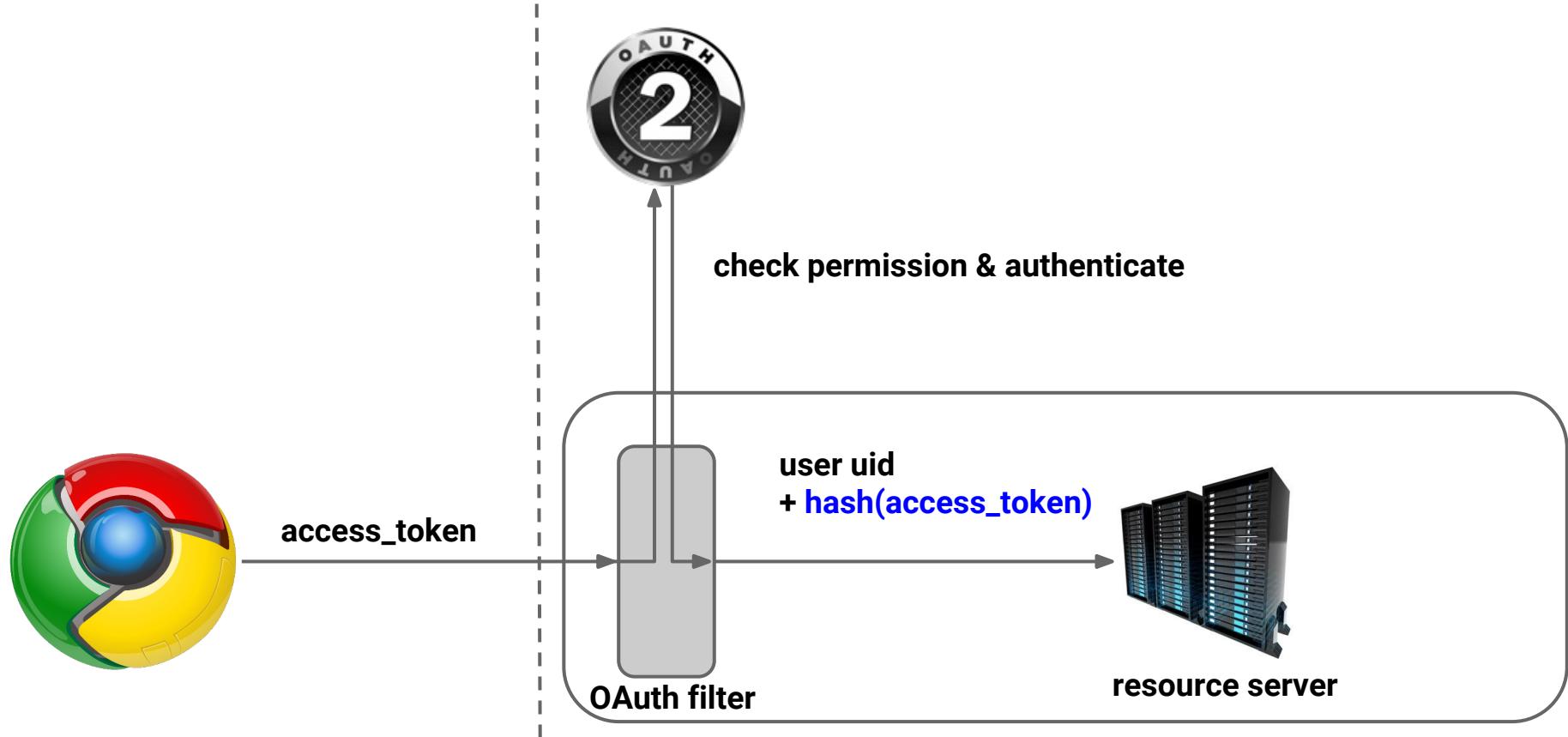


« Pas de bras, pas de chocolat... »

#INTOUCHABLES



# HTTP Request



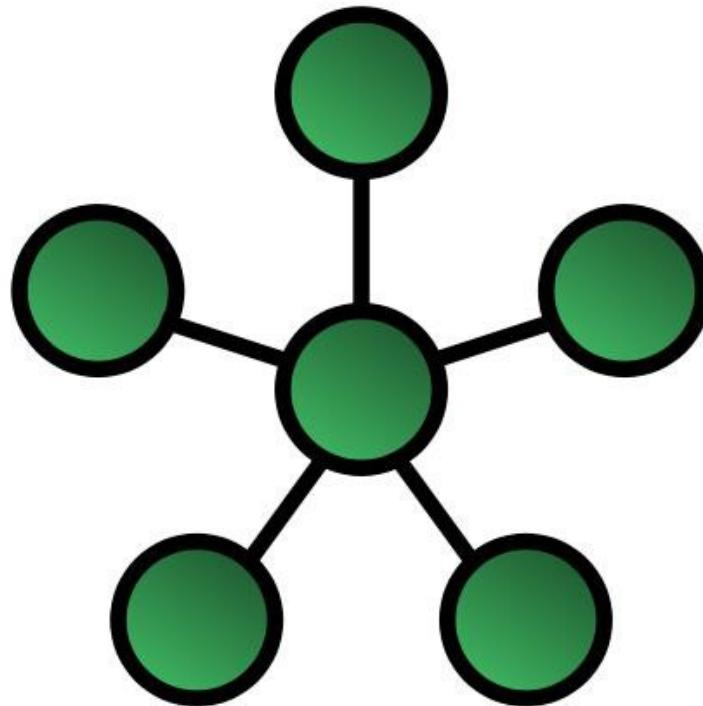


# #TRANSFERT





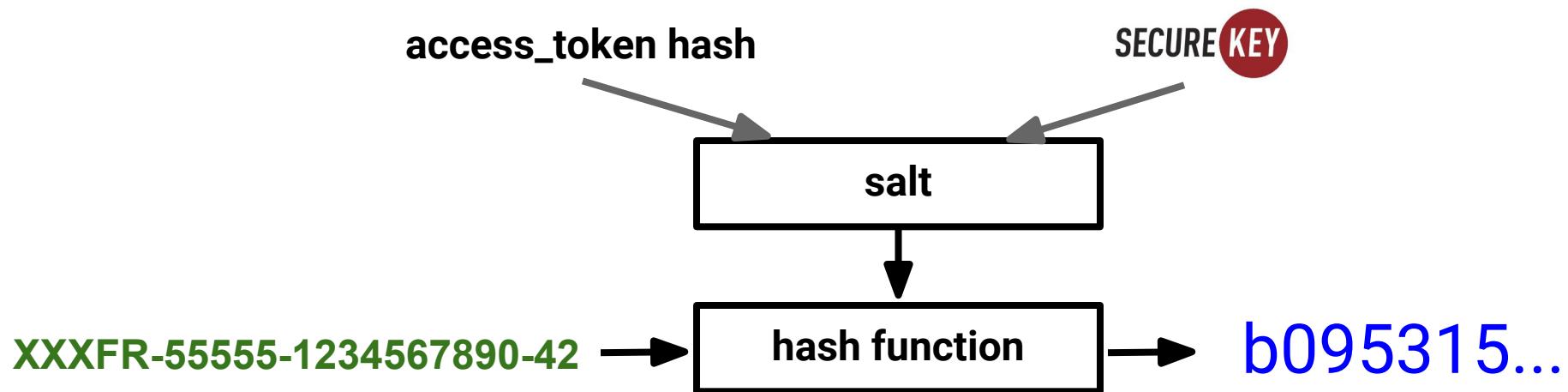
# Accounts availables





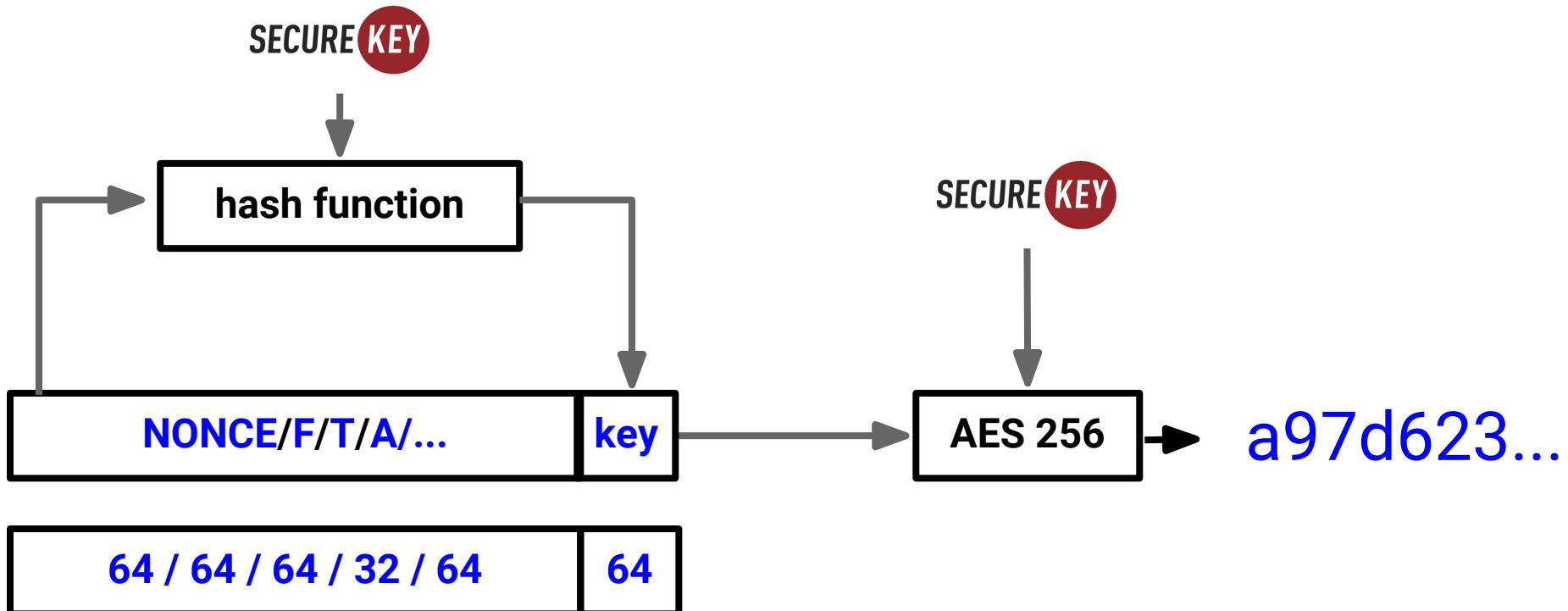
# Accounts availables

```
{  
  '28b7b53...':{... account description},  
  '3bc118d...':{... account description},  
  'b095315...':{... account description},  
}
```



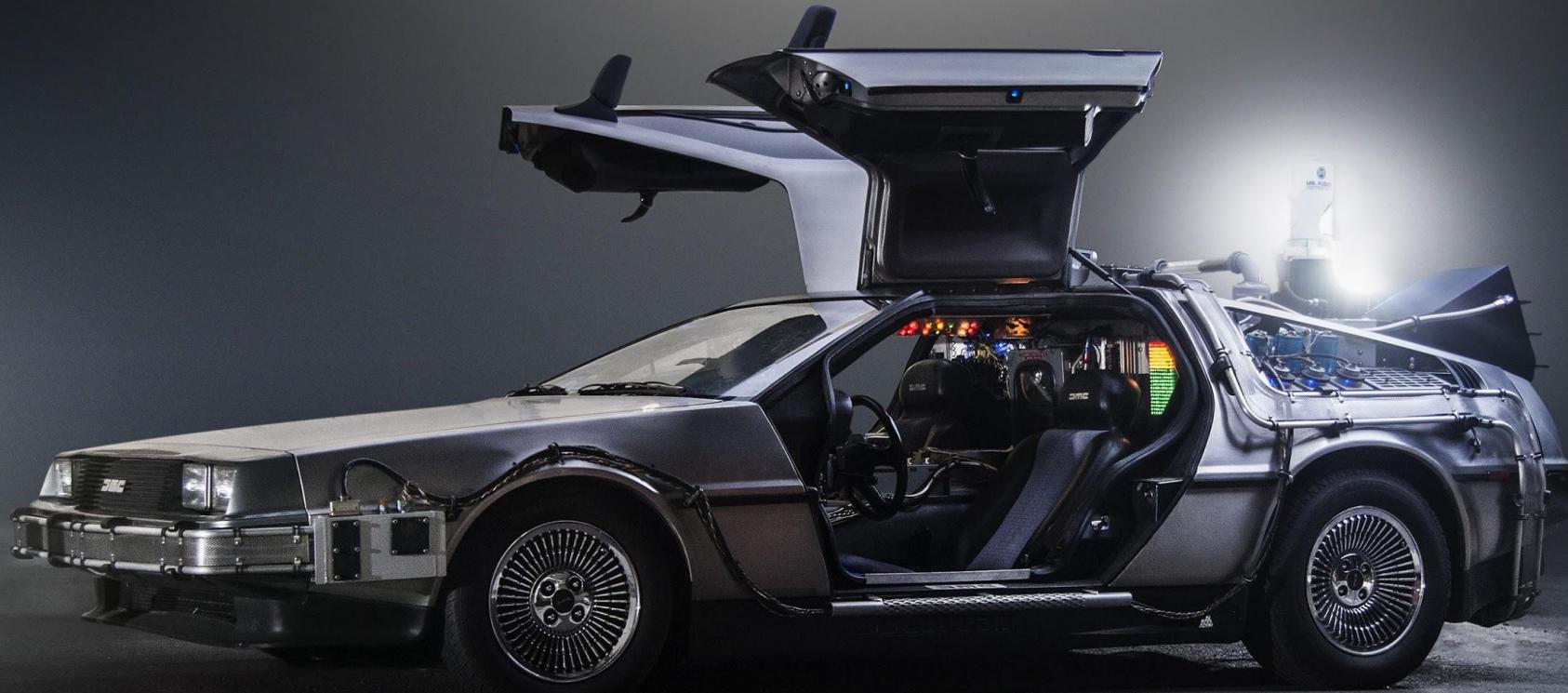


# Authorisation code





# Time machine (TOTP)





# OTP (TOTP) with $d$ digits

$TC = (\text{unixtime}(\text{now}) - \text{unixtime}(T0)) / TS$

$\text{TOTP} = \text{HOTP}(\text{SecretKey}, \text{TC})$

**TOTP-Value** = TOTP mod  $10^d$



# Compute the counter TC

```
byte[] text = new byte[8];
for (int i = text.length - 1; i >= 0; i--) {
    text[i] = (byte) (TC & 0xff);
    TC >>= 8;
}
```



# Hash Mac SHA1 OTP

```
public static byte[] hmac_sha1(byte[] keyBytes, byte[] text) throws... {  
    Mac hmacSha1 = Mac.getInstance("HMAC-SHA-1");  
  
    SecretKeySpec macKey = new SecretKeySpec(keyBytes, "RAW");  
    hmacSha1.init(macKey);  
    return hmacSha1.doFinal(text);  
}
```



# OTP (HOTP) with d digits

$$TC = (\text{unixtime}(\text{now}) - \text{unixtime}(T0)) / TS$$
$$\text{TOTP} = \text{HOTP}(\text{SecretKey}, TC)$$
$$\textbf{TOTP-Value} = \text{TOTP} \bmod 10^d$$



# OTP (TOTP)

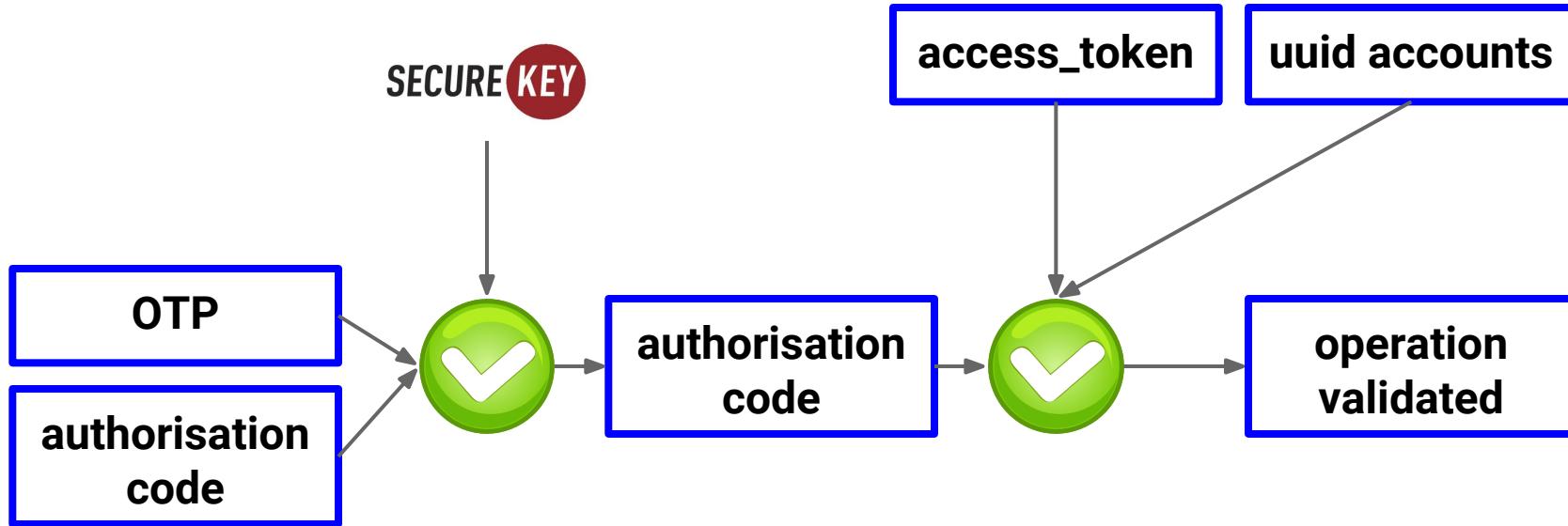
hash(authorisation code)

SECURE 

TOTP = HOTP(**SecretKey**, TC)



# Check transaction





# Benefits

**kill access\_token and sleep deeply**





# Benefits

**Second factor is contextualized**





# Limits: collated by computing

access\_token hash

SECURE 

XXXFR-55555-1234567890-42

28b7b53 → 28b7b53

XXXJA-12125-6662391327-69

3bc118d → 3bc118d

XXXJA-12125-6662391328-54

b095315 100€

XXXJA-12125-6662391329-76

→ ef338a6 ...

XXXJA-12125-6662391330-21

57a1d9f

XXXJA-12125-6662391331-43

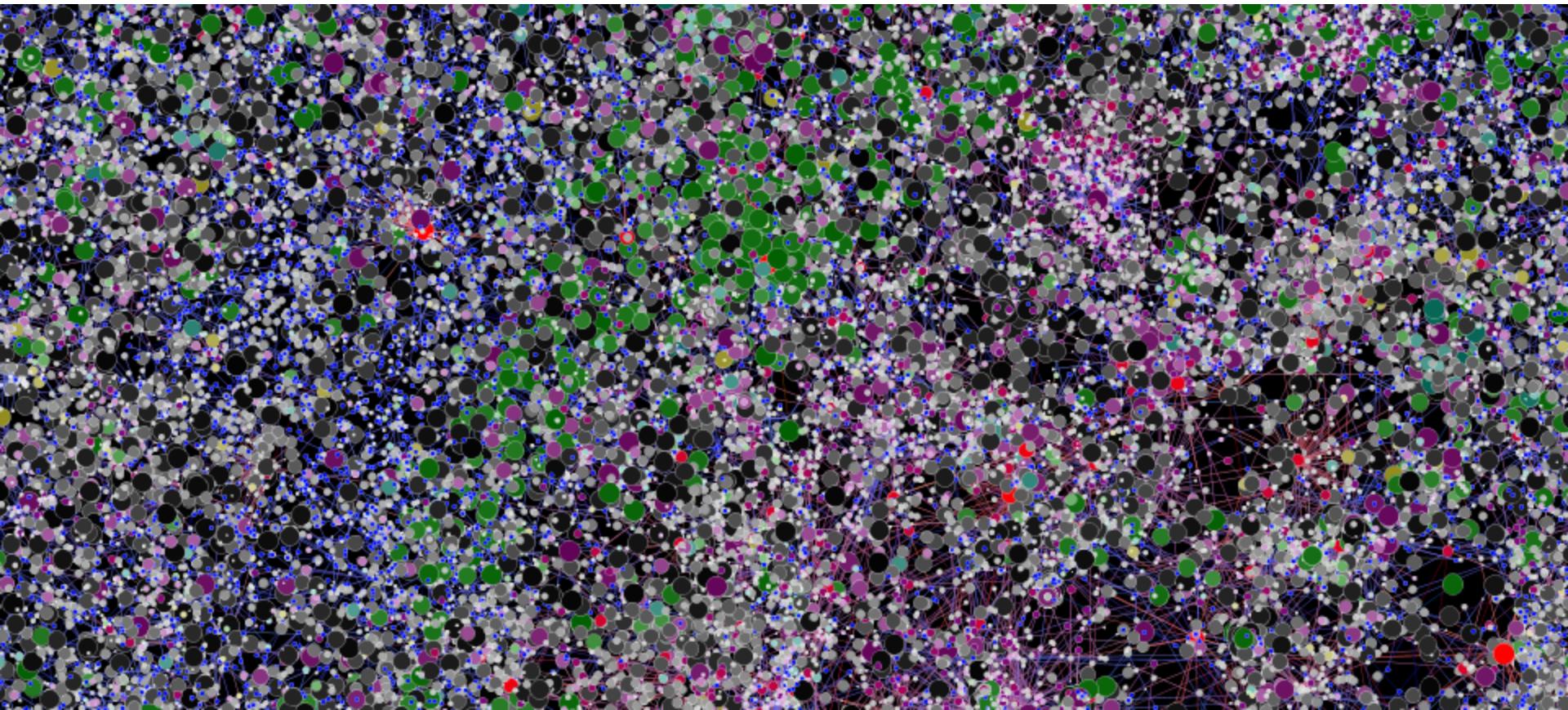
a8e2fc2

XXXJA-12125-6662391332-19

43fc129

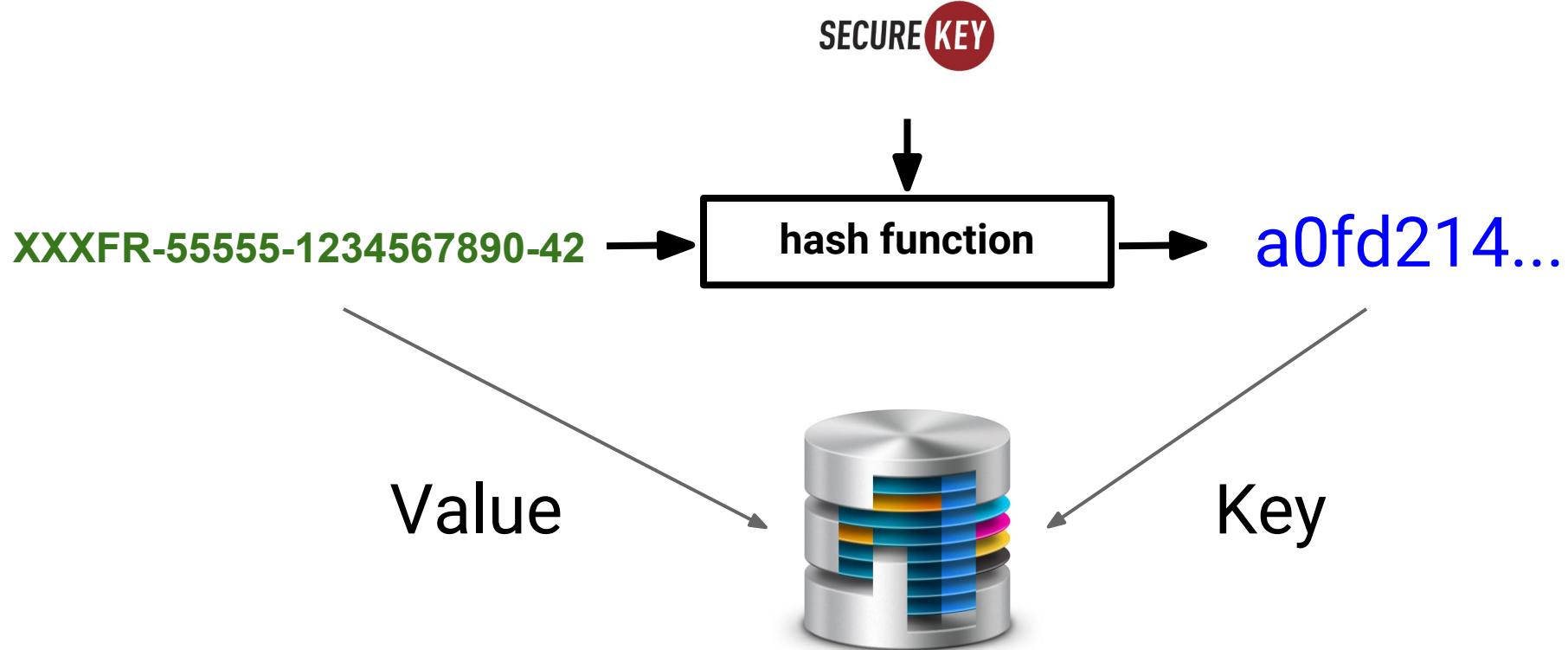


# large entries





k,v





SECURE KEY

access\_token hash

SECURE KEY

hash function

XXXFR-55555-1234567890-42  
a0fd214...

salt

hash function

NONCE / k

key

AES 256

aef7ef19...

64 / 64

64



## #Benefits

**Immutable transaction per user**

**unreplayable**

**only legitim calls to backend**

**internal data structure not exposed**



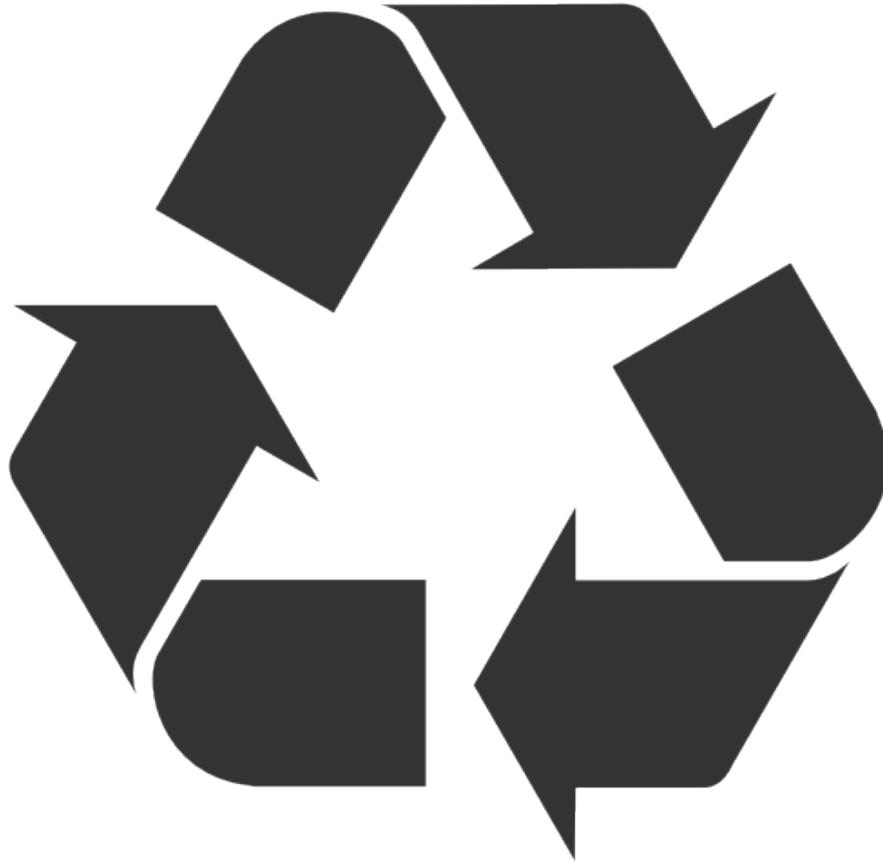
HARD BUT  
GOOD







UNIT CONFIGURED





## Email confirmed



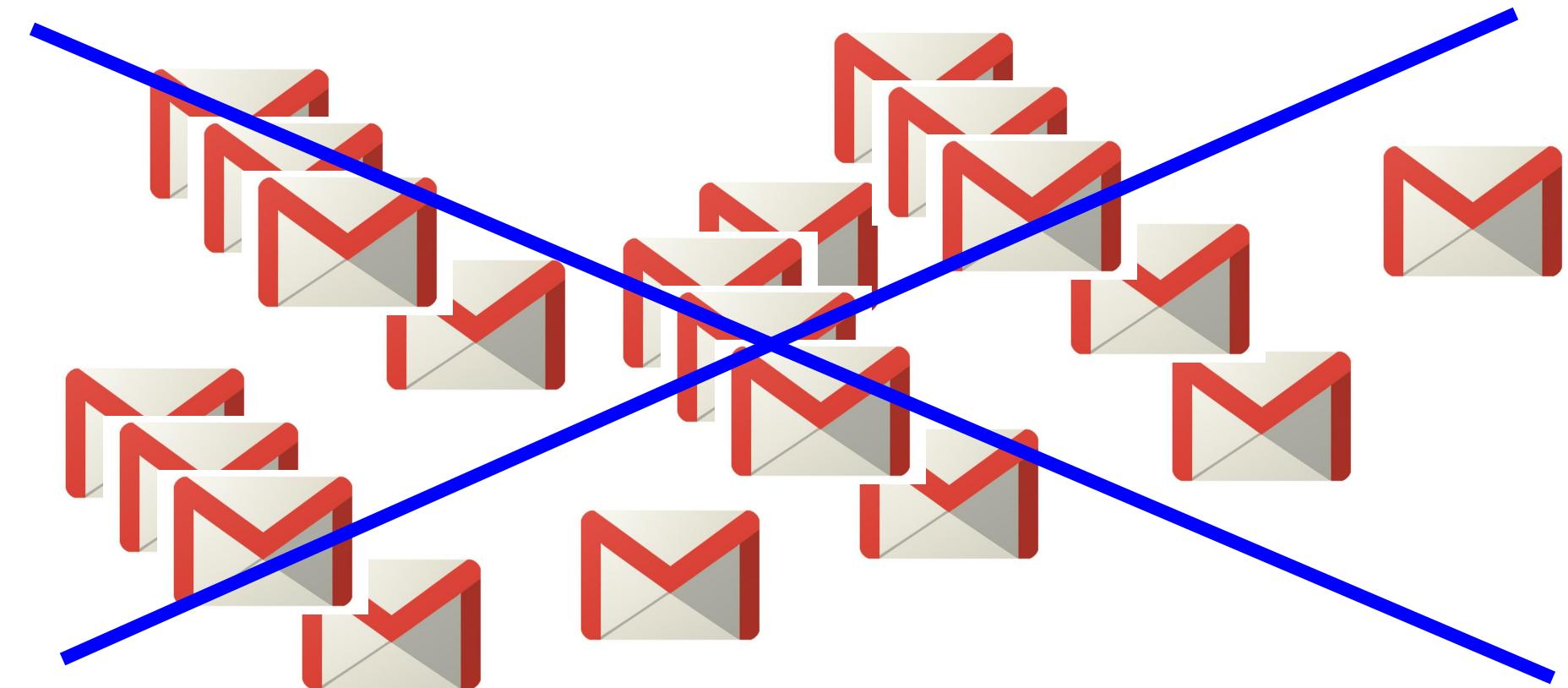
Your email address has been confirmed. To start taking live payments, you should now activate your account.

Later

Activate account











## Email verification



Inbox x

Notifications x

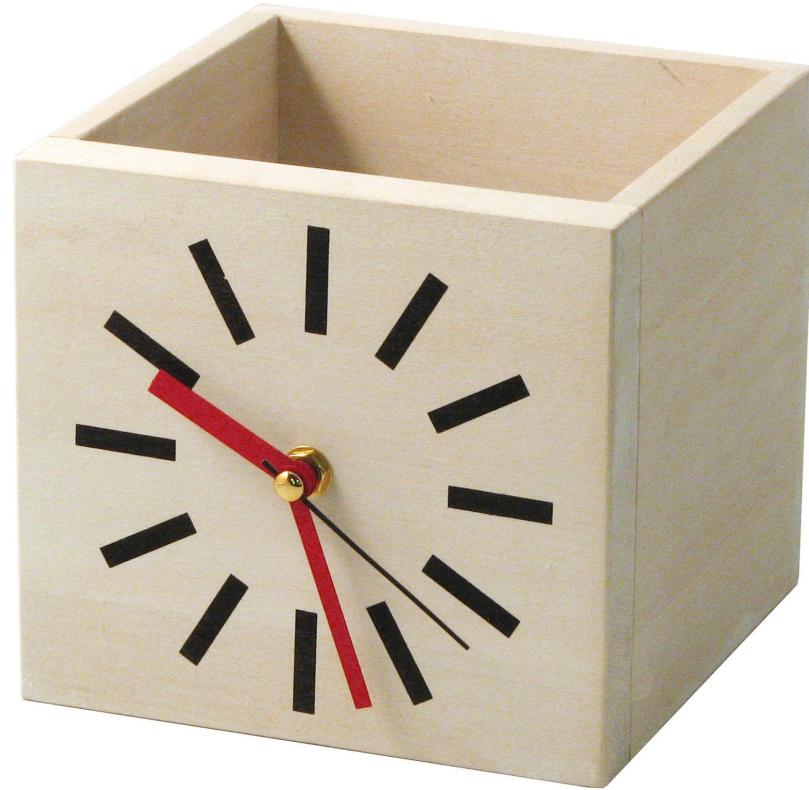


**9lessons Demos** labs@9lessons.info via ar  
to me   11:45 PM (59 minutes ago)      

Hi,

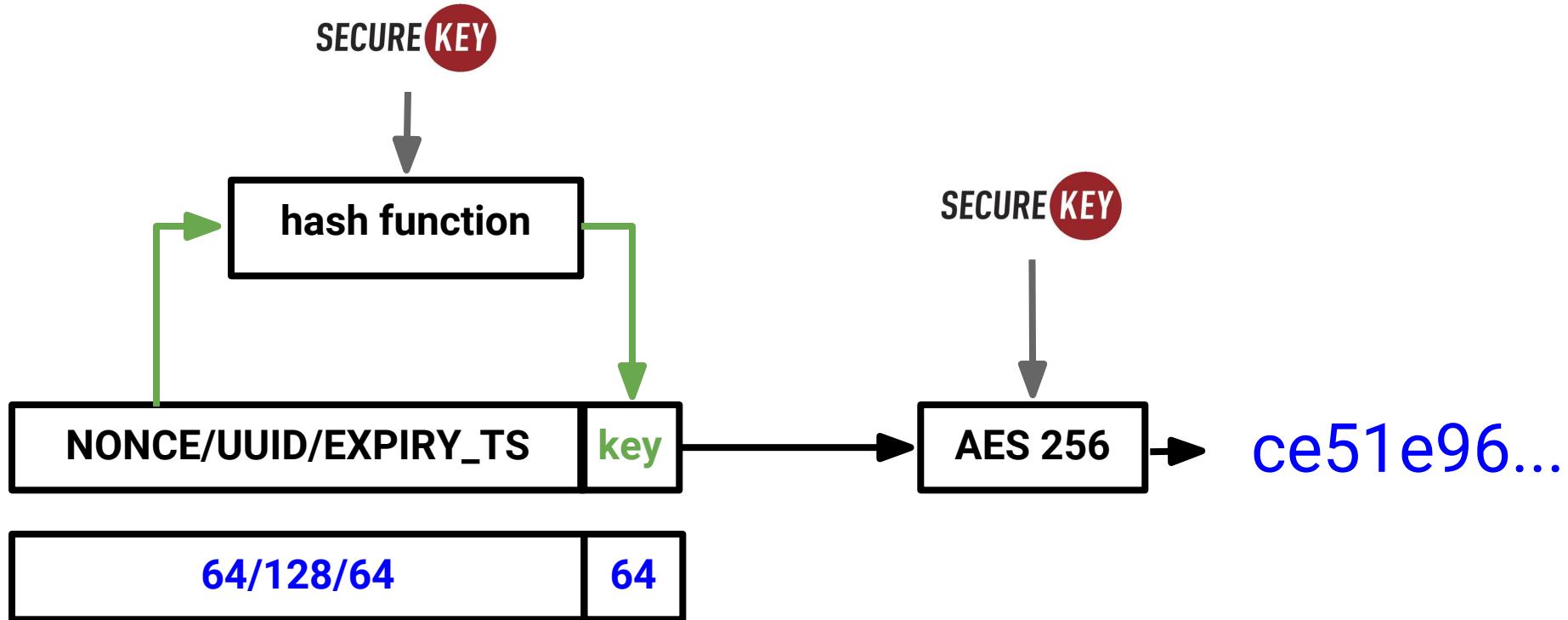
We need to make sure you are human. Please verify your email and get started using your Website account.

[http://demos.9lessons.info/email\\_activation/activation/3272f5b50d0c8ed0bb6361ca844f](http://demos.9lessons.info/email_activation/activation/3272f5b50d0c8ed0bb6361ca844f)





# Activation code





{

‘uuid’: ‘ca793aed-d653-4841-bb90-84e091acedaa’,  
‘issuance’ : 1433600000000,  
**‘activation’: 1433683314198,**

....

}



PAST

Access  
created

NOW

Activation  
Limit

FUTURE





# Optimisation



<b>KEY</b>	ca793aed-d653-4841-bb90-84e091acedaa
<b>VALUE</b>	true
<b>EXPIRY</b>	timestamp expiry date



# Encoding ?





# Compare It

[63, -30, 44, -7, -34, -41, 74, 81]



3fe22cf9ded74a51



P+ls+d7XSIE=





# #Problem

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z a b c d e f g h i j  
k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9 + / =







ANY  
QUESTIONS?  
?