javascript crypto.

ugly duckling with good reason?

whoami

ruby-core

ruby openssl

krypt

freelancer

whoami

germany

i love crypto

i enjoy being the nerd among nerds

i do javascript, too

been doing java even longer

wrote a lot of java crypto code on the job

wrote a lot of ruby & c

crypto code, too

never wrote any javascript crypto code

why?

javascript cryptography

considered harmful

javascriptcrypto

http://www.matasano.com/articles/javascript-cryptography/

!!! client-side !!!

let's see why it's doomed

js served over http

troll-in-the-middle attack

client <----------  ----------> server

serve different files

alter files on the fly

inject <script> tags

etc.

alright, let's take care of the network

https

~~troll-in-the-middle attack~~

client <----------> server

https

mission accomplished ?

troll-on-the-server attack

client <----------> server

works with https, too

client <----------> server

https

if you don't trust them to do

crypto for you

how could you trust

their crypto js code?

so what about signed js?

allows verification of authenticity

still, the same trust issue

easier to set up https to begin with

note:

not a javascript problem per se

tho there are javascript problems as well

crypto people love to fiddle with

bits & bytes

often algorithms require

exact-width integer operations

(e.g. on 8 bit, 32 or 64 bit)

self-fulfilling prophecy:

„i know c, so i design for c“

in their defense:

c is a good choice for
system-level programming

(algorithms need to run on hardware, too)

no out-of-the-box support

for binary data

no out-of-the-box support

for bigintegers

-> workarounds lack native support

-> workarounds are slow

then: browser js problems

lack of a universally supported

„cryptographically secure
pseudo-random
number generator“

(aka csprng)

Math.random is predictable

csprng is at the heart of crypto

without it, crypto becomes a farce

workarounds using mouse movement

and whatnot

meh

clearly something that should be built-in

window.crypto.getRandomValues

lacks wide-spread support

what makes javascript fun

is also what makes javascript crypto hard

dynamic runtime environment

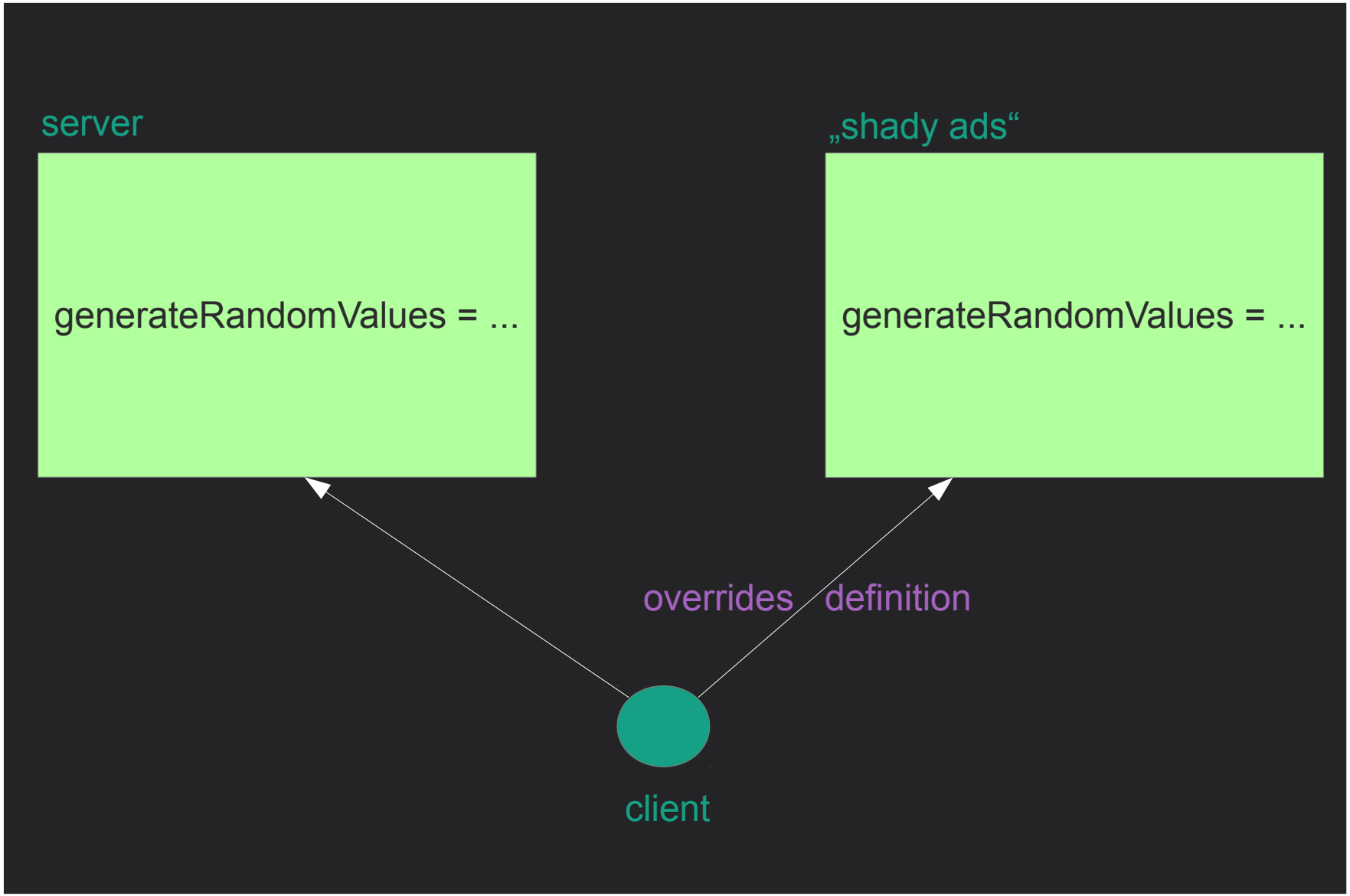
le demo

```
crypto.generateRandomValues = function(array) {
   array[0] = 42;
};
```

doesn't even have to be intentional

javascriptcrypto

```
/**
 * FTFY. I benchmarked it and it's *really* fast.
 */
crypto.encrypt = function(value) {
   /* implement ROT26 algorithm */
   return value;
};
```


server
generateRandomValues = ...
„shady ads“
generateRandomValues = ...
overrides
definition
client

almost impossible

to verify the environment

could be changing continuously

</ whining >

„so javascript crypto sucks,
end of story?“

a new hope

certain things need to stay immutable


sandbox

enable-cors.org

signed javascript

(?)

widely-adopted
browser-built-in

crypto functionality

asm.js

exact-width integer types

= typed array support

(http://www.khronos.org/registry/typedarray/specs/latest/)

bit-fiddling deluxe

fast

w3c web cryptography api

biginteger support

typed array support

built-in csprng

lots of good stuff

„dude, could you
finally get to the point
where you tell us
why we need crypto to begin with?
i never wrote any crypto code
and i certainly don't intend to.“

Q: „would you feel comfortable
sitting on a stranger's lap in the bus?“

why not?

privacy

„if you have

nothing to hide,

you have nothing to fear.“

bullshit

everyone has a secret

that would embarass

them if it were public knowledge

people forgive, but the internet does not

with great power comes great responsibility

but not everybody can be spiderman

while our governments may be

„benevolent in general“

it's individuals that do harm

not the government or „the system“ is evil

people are evil

a series of well-intended myopic decisions

may lead to an evil whole


PRISM

even if companies provide perfect crypto

they store data in plain text

ready for PRISM to pick up

„what can we as individuals
do against things like prism?“

encrypt shit

don't trust applications to do it for you

do it yourself

-> you're gonna need client-side crypto

institutions may have leverage

over a single corp

but not over a billion individuals

surveillance

will tremendously change our path of life

by putting determinism in it

do we want to live in 1984?

germany has a dark history of surveillance

not a single person felt safer

people were in constant fear

you will be told that using crypto is bad

child molesters, drug lords, bla bla

in short:

„crypto is bad

because it will be abused by bad people“

while at the same time this holds true

for their very surveillance measures

there is nothing with a light side

that doesn't also have a dark side

except maybe

this guy

excursion. homomorphic encryption.

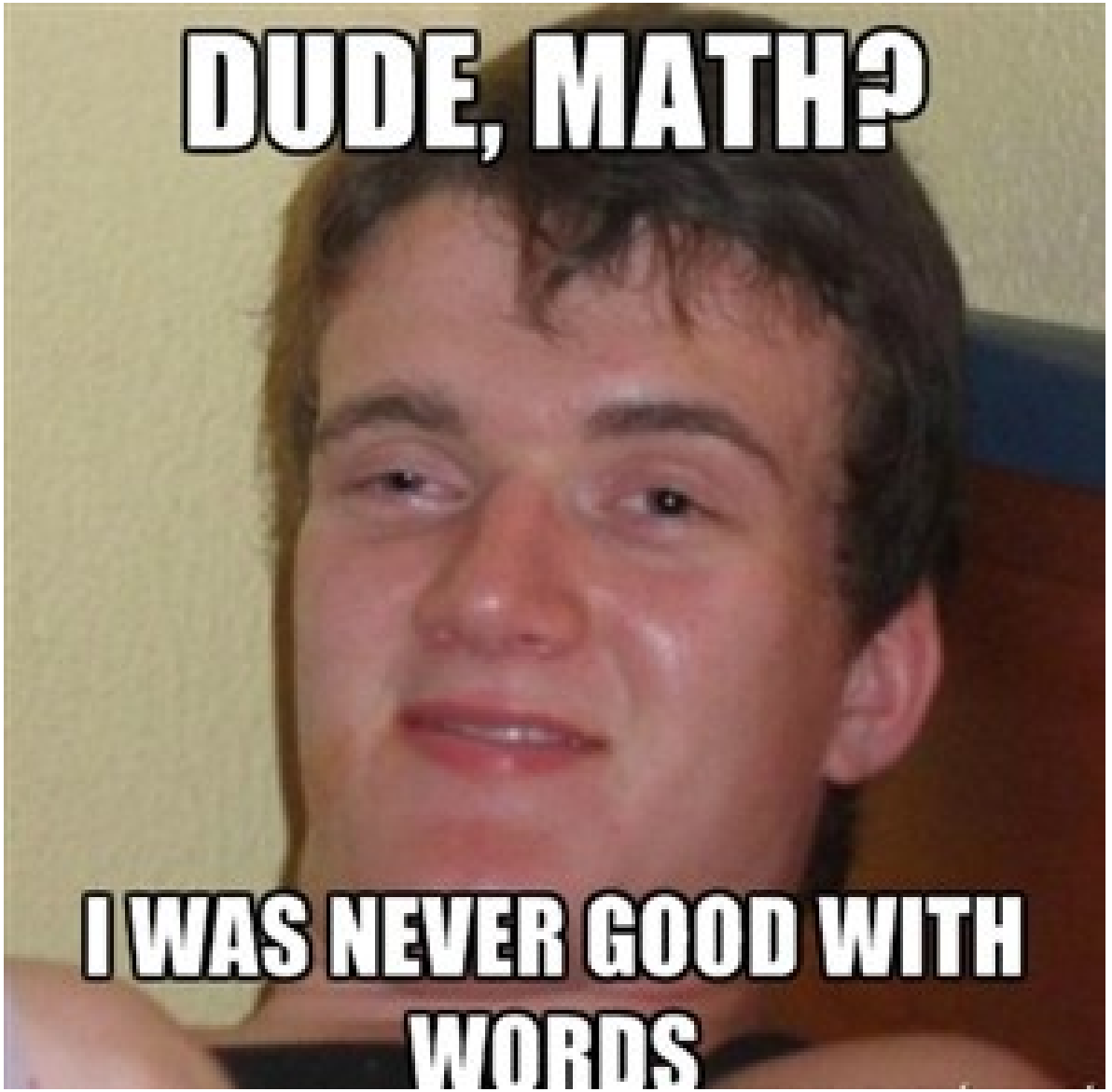
homomorphism

$f: (G,*) \rightarrow (H,*')$

such that

$f(g1 * g2) = f(g1) *' f(g2)$

for any elements $g1, g2 \in G$.


DUDE, MATH?
I WAS NEVER GOOD WITH
WORDS

$f(x) := 1/x$

G, H := real numbers

$*, *'$:= multiplication

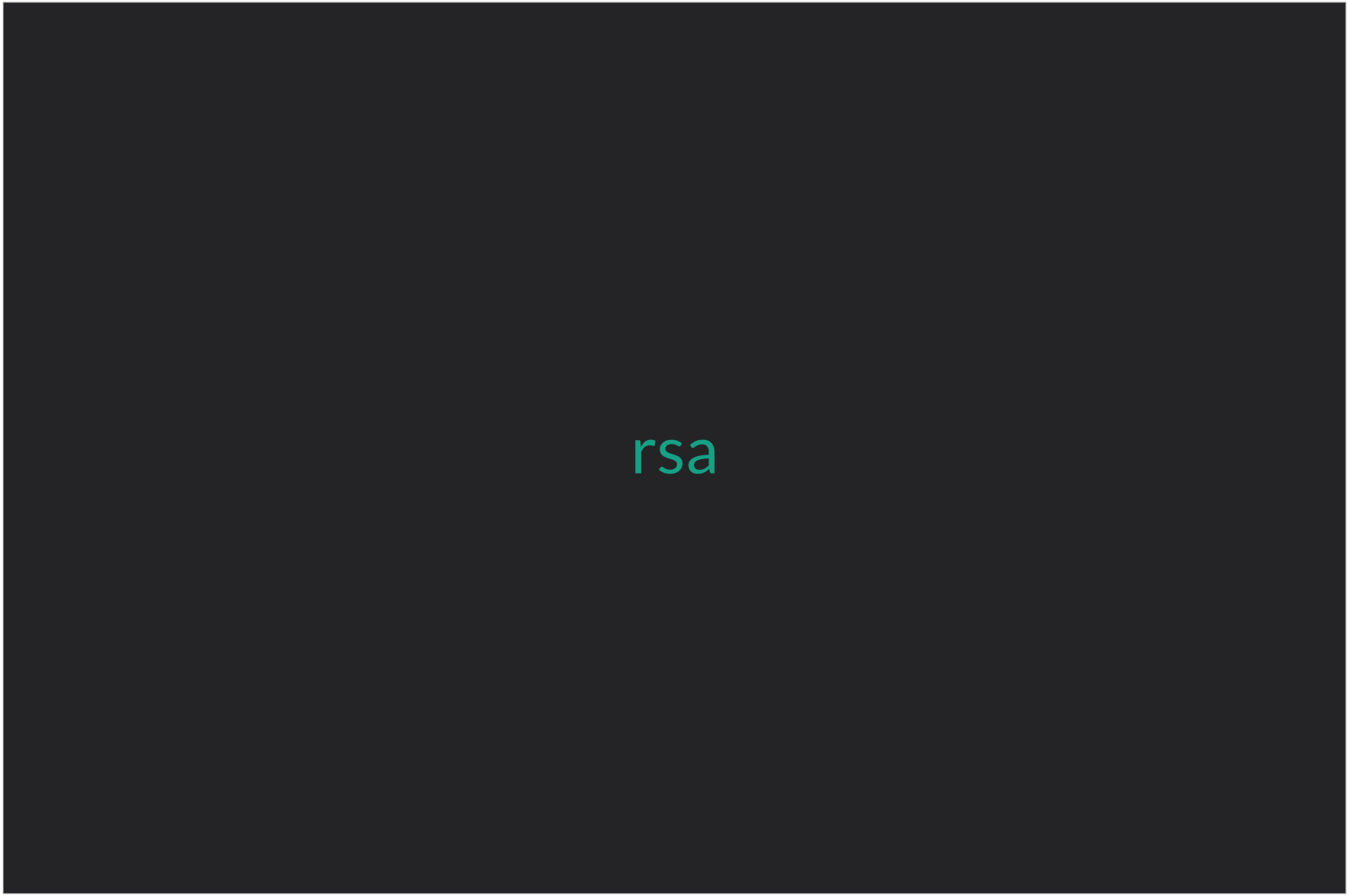
$$f(a * b)$$

$$= 1 / (a * b)$$

$$= 1/a * 1/b$$

$$= f(a) *' f(b)$$

„group homomorphism“


rsa

public key e

modulus m

encryption E(x) := x^e mod m

$$E(x1*x2)$$

$$= (x1 * x2)^e \bmod m$$

$$= x1^e * x2^e \bmod m$$

$$= x1^e \bmod m * x2^e \bmod m$$

$$= E(x1) * E(x2)$$

but

$$E(x1+x2)$$

$$= (x1+x2)^e \bmod m$$

~~$$= x1^e + x2^e \bmod m$$~~

relatively easy to find

homomorphisms

for one of the two operations

homomorphism in both operations:

fully homomorphic encryption

$f: (G,+,*) \rightarrow (H,+',*')$

such that

$f(g1 * g2) = f(g1) *' f(g2)$

and

$f(g1 + g2) = f(g1) +' f(g2)$

for any elements $g1, g2 \in G$.

why is this so desirable?

it preserves the ring structure, that's why!

layman terms:

perform algorithms
on
encrypted data

think:

google executes your search
and returns the correct result
without learning anything
about your search term

yay, porn

holy grail of privacy

craig gentry

(http://www.americanscientist.org/issues/pub/2012/5/alice-and-bob-in-cipherspace)

„ok, cool story, bro,
but would you mind
telling us what we can
do right now?“

w3c web crypto api

major step forward

will let you do cool things like:

keep your data private in the cloud

store stuff securely in your

offline html5 app

log into applications

without revealing the actual password

cryptographically sign documents

in a browser context

secure messaging

strong authentication using

smart cards or other tokens

(goodbye, java applets!)

„wait a minute -
didn't you say that we
cannot trust js code
anyway? so how is the
w3c api gonna help?!“

it all boils down to trust

at some point, you need to trust

thinking this further:

how do you know

any of your software is authentic and/or benevolent?

chicken & egg

https download/verifying signature ->

need software for that ->

infinite recursion

still, major improvement

but:

encrypting data with aes

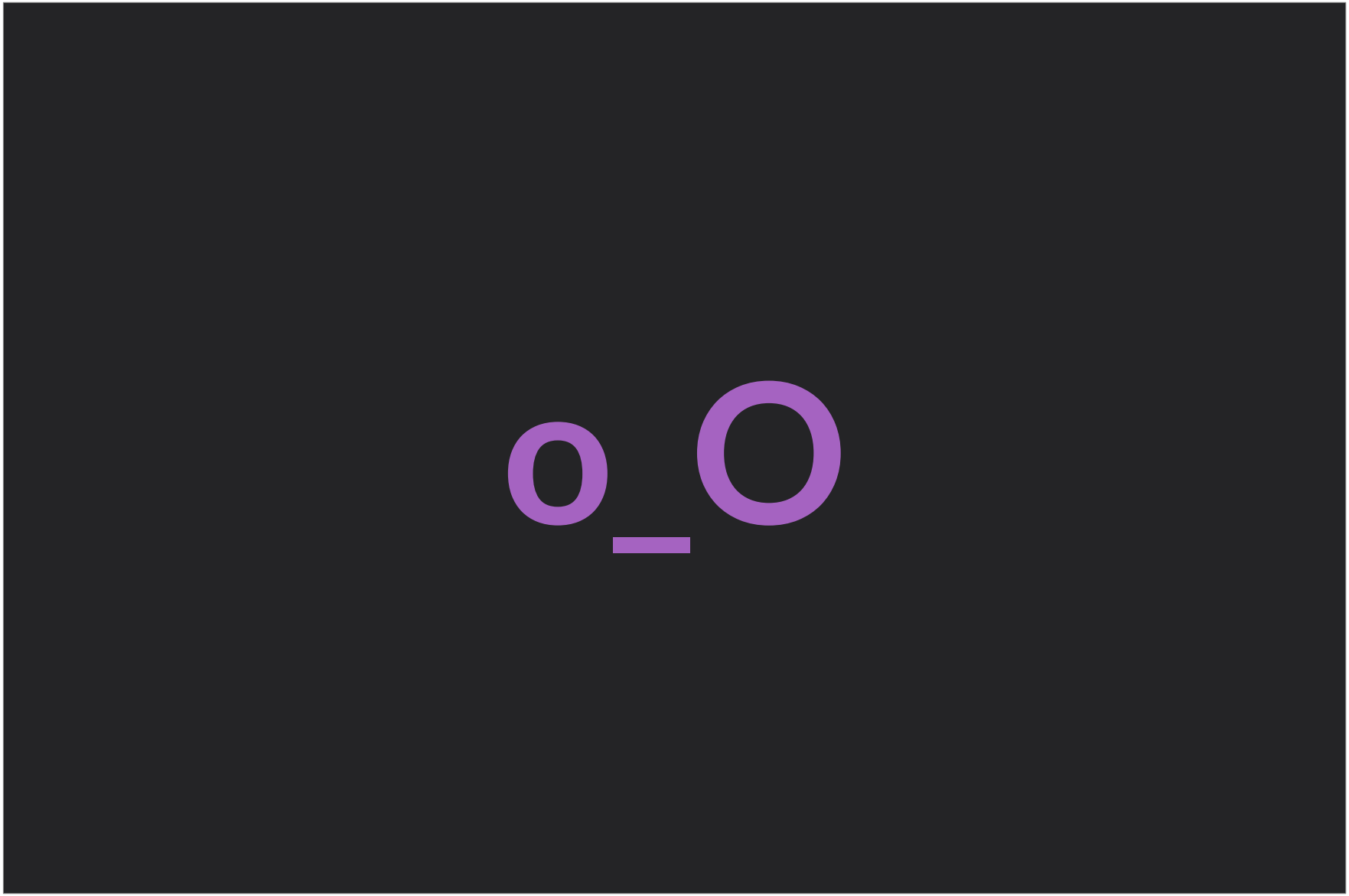
```
var data = „Le secret“;
var clearDataArrayBufferView = convertPlainTextToArrayBufferView(data);

var aesAlgorithmKeyGen = {
  name: "AES-CBC",
  params: { length: 128 }
};

var aesAlgorithmEncrypt = {
  name: "AES-CBC",
  params: { iv: window.crypto.getRandomValues(new Uint8Array(16)) }
};

var cryptoKeyGen = window.crypto.generateKey(aesAlgorithmKeyGen,
                                             false,
                                             ["encrypt"]);

cryptoKeyGen.oncomplete = function(event) {
  var aesKey = event.target.result;
  var aesOp = window.crypto.encrypt(aesAlgorithmEncrypt,
                                    aesKey,
                                    clearDataArrayBufferView);
  aesOp.oncomplete = function(event) {
    var ciphertext = event.target.result;
  };
  aesOp.onerror = function(event) { console.error("Unable to encrypt."); };
};
```


o_O

low-level api in the tradition of openssl

with full control

but also full possibility to hang yourself

reasons

stay compatible with legacy apps

easily implemented -

just wrap the underlying c library

no secure defaults

what i want is this

javascriptcrypto

```
var data = „Le secret“;
var key = window.crypto.generateKey();
var encrypted = window.crypto.encrypt(key, data);
/* nuff said */
```

crypto is hard, sure

but do crypto apis have to be, too?

krypt. semper pi.

framework that wraps expert apis™

to make crypto accessible for human beings

so how would you like the sound of


KRYPT
SEMPER
platform- and library-independent Cryptography for Ruby

krypt.js

thank you

https://github.com/krypt

http://martinbosslet.de

martin.bosslet@gmail.com

@_emboss_