SUDHAUS7

ConFoo.CA
DEVELOPER CONFERENCE

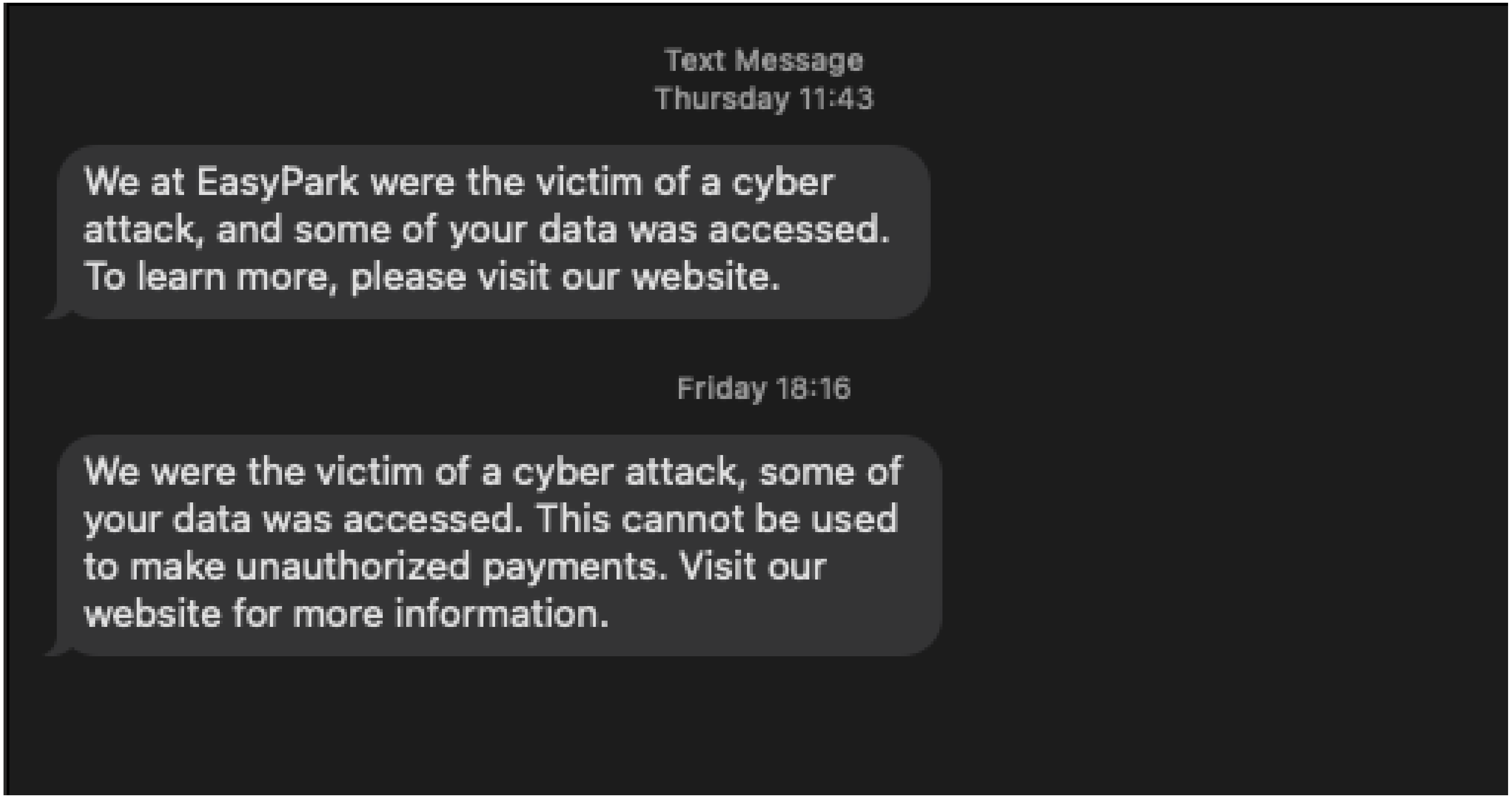
A PRACTICAL INTRODUCTION TO ENCRYPTION IN PHP

CONFOO MONTREAL 2024

FRANK BERGER

A BIT ABOUT ME

- Frank Berger
- Head of Engineering Sudhaus7, a label of B-Factor GmbH, Stuttgart, Germany
- Started as an Unix Systemadministrator who also develops in 1996
- Web development since 2000
- TYPO3 since 2005

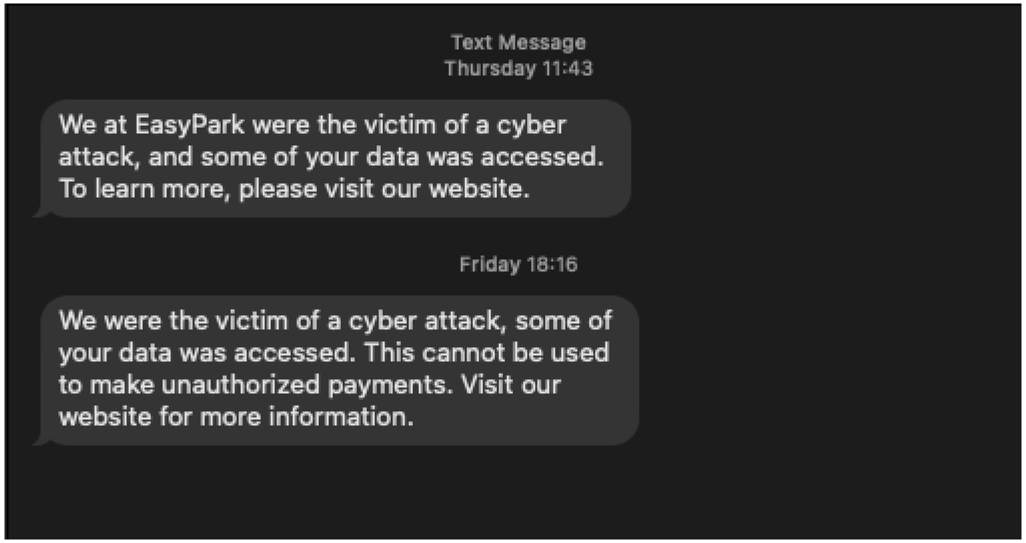


Text Message
Thursday 11:43

We at EasyPark were the victim of a cyber attack, and some of your data was accessed. To learn more, please visit our website.

Friday 18:16

We were the victim of a cyber attack, some of your data was accessed. This cannot be used to make unauthorized payments. Visit our website for more information.


Text Message
Thursday 11:43

We at EasyPark were the victim of a cyber attack, and some of your data was accessed. To learn more, please visit our website.

Friday 18:16

We were the victim of a cyber attack, some of your data was accessed. This cannot be used to make unauthorized payments. Visit our website for more information.

I have questions!!

Why was my data not encrypted?

Who else had access to my data?

SUDHAUS7

ConFoo.CA
DEVELOPER CONFERENCE

WHAT THIS TALK IS NOT ABOUT

How encryption algorithms (like RSA) work

or how hashing algorithms work

or which one is the best

or differences between encryption methods

We just accept that there are hashing algorithms and encryption methods that are proofen to work and we're going to use

WHAT THIS TALK IS ABOUT

How to start encrypting stuff in PHP

using what PHP gives us

with asymmetric encryption (Private and Public Keys)

to give a practical primer how to handle encrypted data

SUDHAUS7

ConFoo.CA
DEVELOPER CONFERENCE

ASYMMETRIC ENCRYPTION

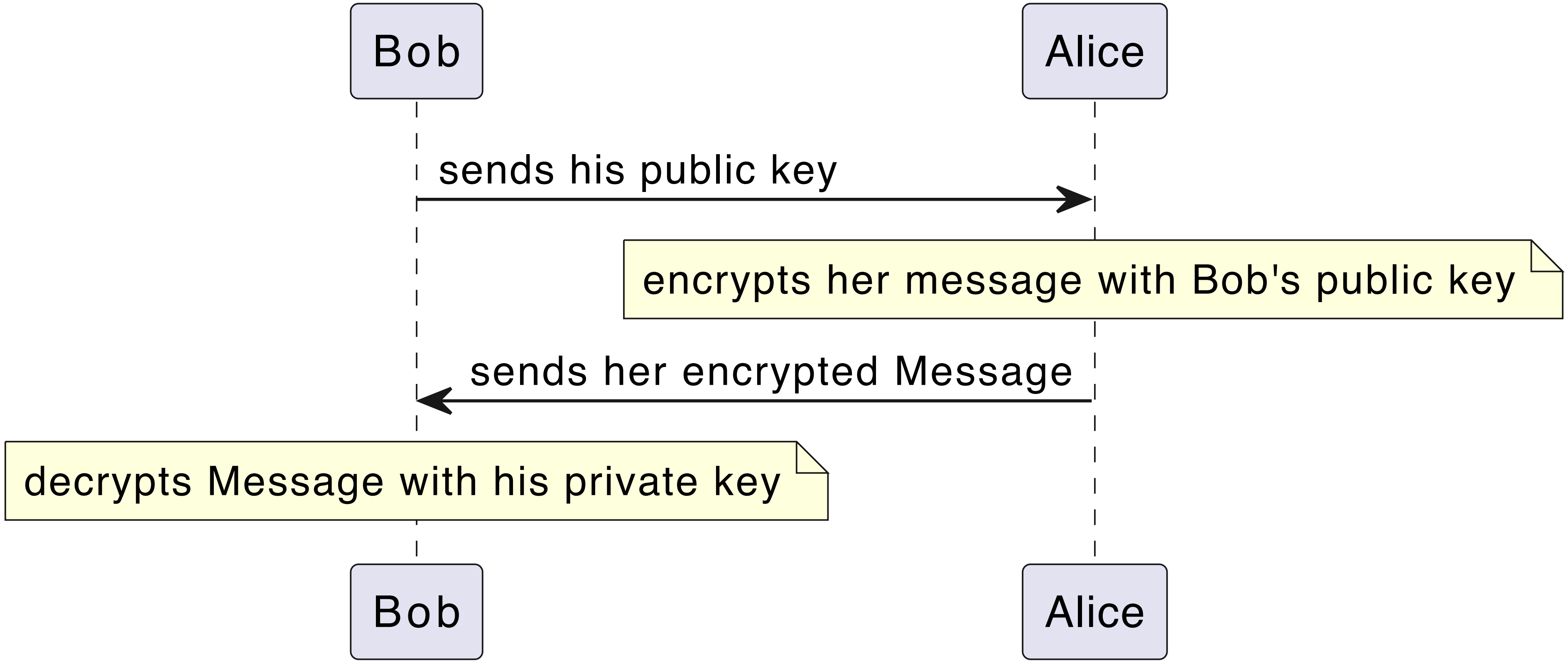
Public key encrypts

Private Key decrypts

The public key can be derived from the private key

The private key is only known to you!
The public part can be printed on a billboard

This is what SSH, SSL, TLS, PGP and many more are doing


Bob
Alice
sends his public key
encrypts her message with Bob's public key
sends her encrypted Message
decrypts Message with his private key
Bob
Alice

2 Major Crypting Libraries **native** in PHP

OpenSSL

Sodium (libsodium implementing NaCl)

Additionally PGP/OpenGPG (PECL)

(there are more written in PHP itself)

We need to choose a hashing algorithm

Dozens of encryption algorithms

Which one do we choose?

We're going to use OpenSSL and use AES-256-CBC for now (and SHA256 for hashing)

Check this function

```
OpenSSL_get_cipher_methods()
```

to find out which algorithms are available

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

creating a key (in OpenSSL)

```
$keyResource = OpenSSL_pkey_new([
  "digest_alg" => "sha256",
  "private_key_bits" => 4096,
  "private_key_type" => OpenSSL_KEYTYPE_RSA,
]);
OpenSSL_pkey_export($keyResource, $private_key);
$public_key = OpenSSL_pkey_get_details($keyResource)["key"];
```

to secure the private key with a password export it with one

```
OpenSSL_pkey_export($keyResource, $private_key,'secretpassword');
```

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

This is what such a key looks like

```
-----BEGIN PUBLIC KEY-----
MIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEAvffa64mFV3UKf6G6OLlQ
olZxQZsdCuRtEVFFqXCbtOKSFWsfXuZ+hYqoWFUrTGdApKV5hdGm9aXbcSzs3d8J
PjGF7uZUoiQvoasdCi/9Y16W1nxYeLXZOjLlHxLTzWEee2XGketBjBEpYQ8jSyaH
LeCMHc/cxgfCfJNR6HD/ZzQpt65Ace+0LpkwxZ6Y0Sp3cnO0vdIVK9Fzpgm7Dz9W
FAhq7e+43CyLAT+vAl9LUN3BmSnDD2bbeQ40BnGJ0FlQKTxLjt0DkMswSQDpyyDT
H5Ni/Sq6hh5hXLuPmwGkY1VnW/UE8mHYuDNACtMyHQVBhv1oz7r8dObajm1f71lW
JAMVU95yHfKKtx2cUXr1iah9L8Cxdlh9pPBUWYwEOHBPvsP48IPPpjSdkLPOEqrp
gXs+YREjEeMOZGWIZ+9+0hno8w7nfCnExQtMpA1lkvOpaBKf4J22T+rFeHOyESB8
OwfqhaMzPGbmkAKw8IcKwMB4ba4n+ZQbepK80XWBH5valh6TBH5YYT6zDLgUXgIf
UCiH/toh50xPpJmXf3it3BrsicNnFSpcSLKlFnVVy6NoCa/rY4NbQGbxiNMWrOfT
Cl9i8APoC38mxXOq3ezjJplSnymX9lJDuvGN5mURM7cwFDtylhp5V4+pZZ+eayRF
TADumyJqEo/pxKPqLkkAiJECAwEAAQ==
-----END PUBLIC KEY-----
```

The private key has a different header and footer and is much longer

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

Simple encryption / decryption

```php
$iv_length = OpenSSL_cipher_iv_length( 'aes-256-cbc' );
$iv   = OpenSSL_random_pseudo_bytes( $iv_length );
$pub = OpenSSL_pkey_get_public( $public_key );
OpenSSL_seal(
  'hello confoo',
  $sealed,
  $encrypted_keys,
  [$pub],
  'aes-256-cbc',
  $iv
);
OpenSSL_free_key($pub);

$private = OpenSSL_pkey_get_private( $private_key );
OpenSSL_open($sealed,$opened,$encrypted_keys[0],$private,'aes-256-cbc
echo $opened,"\n";
```

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

Simple encryption / decryption

```php
$iv_length = OpenSSL_cipher_iv_length( 'aes-256-cbc' );
$iv   = OpenSSL_random_pseudo_bytes( $iv_length );
$pub = OpenSSL_pkey_get_public( $public_key );
OpenSSL_seal(
  'hello confoo',
  $sealed,
  $encrypted_keys,
  [$pub],
  'aes-256-cbc',
  $iv
);
OpenSSL_free_key($pub);

$private = OpenSSL_pkey_get_private( $private_key );
OpenSSL_open($sealed,$opened,$encrypted_keys[0],$private,'aes-256-cbc
echo $opened,"\n";
```

PROBLEM: ONLY ONE PERSON CAN ACCESS THE DATA

Think different actors who need to access a dataset

Solution: we could encrypt the data multiple times

Problem: large data gets even larger

Problem: multiple recipient channels

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

SOLUTION

In a nutshell: we create a random key for every dataset and encrypt the dataset symmetrically once!

Then we encrypt that generated key asymmetrically with each public key that needs to access the data-set

Then we store the created fragments with the encrypted dataset.

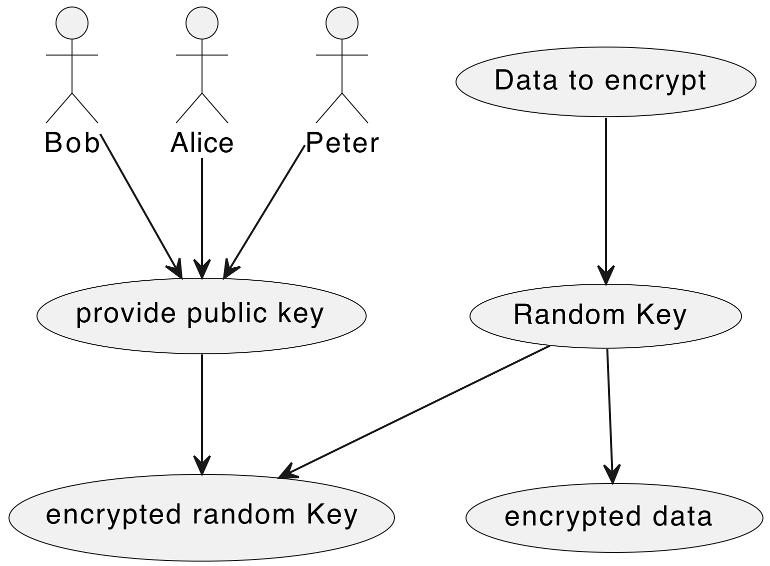
An actors' private key can then be used to de-crypt the random key, which in turn can de-crypt the dataset.

this is what OpenSSL_seal and OpenSSL_open does

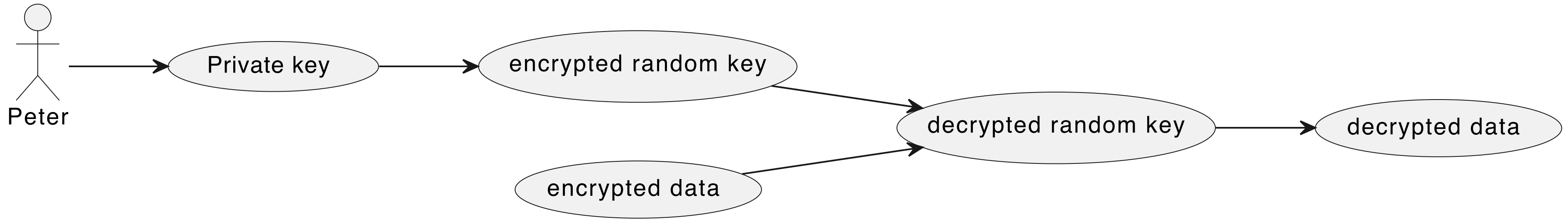
SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

WORKFLOW - STEP 1



WORKFLOW - STEP 2



```php
$public_keys = [
  $pubBob, // OpenSSL_pkey_get_public( $Bob_publickey);
  $pubAlice, // OpenSSL_pkey_get_public( $Alice_publickey);
  $pubPeter // OpenSSL_pkey_get_public( $Peter_publickey);
];
$iv_length = OpenSSL_cipher_iv_length('aes-256-cbc');
$iv  = OpenSSL_random_pseudo_bytes($iv_length);
$message = 'hello confoo 2024';
OpenSSL_seal(
  str_pad( $message, strlen($message)+16-strlen($message)%16,"\0"),
  $sealed,
  $encrypted_keys,
  $public_keys,
  'aes-256-cbc', $iv
);
```

```php
$public_keys = [
  $pubBob, // OpenSSL_pkey_get_public( $Bob_publickey);
  $pubAlice, // OpenSSL_pkey_get_public( $Alice_publickey);
  $pubPeter // OpenSSL_pkey_get_public( $Peter_publickey);
];
$iv_length = OpenSSL_cipher_iv_length('aes-256-cbc');
$iv   = OpenSSL_random_pseudo_bytes($iv_length);
$message = 'hello confoo 2024';
OpenSSL_seal(
  str_pad( $message, strlen($message)+16-strlen($message)%16,"\0"),
  $sealed,
  $encrypted_keys,
  $public_keys,
  'aes-256-cbc', $iv
);
```

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

```php
$public_keys = [
  $pubBob, // OpenSSL_pkey_get_public( $Bob_publickey);
  $pubAlice, // OpenSSL_pkey_get_public( $Alice_publickey);
  $pubPeter // OpenSSL_pkey_get_public( $Peter_publickey);
];
$iv_length = OpenSSL_cipher_iv_length('aes-256-cbc');
$iv   = OpenSSL_random_pseudo_bytes($iv_length);
$message = 'hello confoo 2024';
OpenSSL_seal(
  str_pad( $message, strlen($message)+16-strlen($message)%16,"\0"),
  $sealed,
  $encrypted_keys,
  $public_keys,
  'aes-256-cbc', $iv
);
```

```php
$private = OpenSSL_pkey_get_private( $Peter_privatekey );
OpenSSL_open(
  $sealed,
  $opened,
  $encrypted_keys[2], // we know Peters' was the third key
  $private,
  'aes-256-cbc',$iv
);
echo $opened,"\n"; // my secret message
```

SUDHAUS7

ConFoo.CA
DEVELOPER CONFERENCE

Things we need to keep track of:

1. $sealed - the sealed data
2. $iv - initialization vector (if the algorithm uses it)
3. $encrypted_keys - the random key that has been created
4. the encryption method
5. Which public key belongs to which encrypted key

SUDHAUS7

ConFoo.CA
DEVELOPER CONFERENCE

LETS ENCRYPT

```php
$message = 'Bonjour ConFoo 2024!!';

OpenSSL_seal(
  str_pad( $message, strlen($message)+16-strlen($message)%16,"\0"),
  $sealed_data, $encrypted_keys,
  $public_keys, $method, $iv
);

$final_encrypted_keys = [];
foreach($public_keys as $idx=>$key) {
  $checksumm = sha1(OpenSSL_pkey_get_details($pub)['key']);
  $final_encrypted_keys[$checksumm]=$encrypted_keys[$idx];
}
// $dataset can be stored
$dataset  = base64_encode(serialize([
  $method, $iv, $final_encrypted_keys, $sealed_data
]));
```

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

PETER WANTS TO READ

```php
$private = OpenSSL_pkey_get_private( $peters_privatekey);
$public = OpenSSL_pkey_get_details($private)["key"];

$serialized_data = base64_decode($dataset);
[$method,$iv,$encrypted_keys,$sealed] = unserialize($serialized_data)

OpenSSL_open(
  $sealed,
  $opened,
  $encrypted_keys[sha1($public)],
  $private, $method, $iv
);
echo $opened;
```

DIFFERENCES WITH SODIUM

Sodium **derives** the keypair from a passphrase

Sodium does not have a comparable seal/open function with multiple public keys like OpenSSL

Sodium is (more?) secure than OpenSSL against memory/timing attacks

Sodium is simpler (for the programmer) to use than OpenSSL - and fast

Not all languages have sodium support

SO FAR SO GOOD

This was the easy part..

SUDHAUS7

ConFoo.CA
DEVELOPER CONFERENCE

QUESTIONS YOU NEED TO ASK WHEN YOU WANT TO START ENCRYPTING DATA

Who needs to access the data?

On what platforms / tools?

What (programming) languages?

How will we do the key management?

This will create your threat model..

KEYMANAGEMENT IN YOUR APPLICATION?

Create Keys for every User?

Store the Private and the Public key?

Secure the Private key with the users passwort (?)

Will you have a master key?

CONSIDER OPENGPG/PGP

Can be used for E-Mails and Messaging

Public-key servers are available

Implementations in several programming languages

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

SUMMARY

This is not an ultimative guide, you have to inform yourself

Some encryption is better than no encryption

Beware of false security!

If your security / threat model means you can not show your code, it is not secure by definition!

IN THE WORKS (SHAMELESS PLUG)

Guard7

a TYPO3 extension and framework

SUDHAUS7

ConFoo.CA DEVELOPER CONFERENCE

THANK YOU
I AM HERE FOR THE REST OF THE WEEK

Github with the examples:
https://github.com/codeseveneleven/talk-intro-to-encryption

X: @FoppelFB

Fediverse: @foppel@phpc.social

https://sudhaus7.de/

fberger@sudhaus7.de

SUDHAUS7

ConFoo.CA

DEVELOPER CONFERENCE