Variability Modeling of Cryptographic Components

Sarah Nadi

FOSD 2015 Meeting — May 14, 2015

Joint work with: Karim Ali, Steven Arzt, Eric Bodden,
Sebastian Erdweg, Steffan Krüger, and Mira Mezini

CROSSING

TECHNISCHE UNIVERSITÄT DARMSTADT

Data is Valuable

Data is Valuable

Sony Pictures, 2014
(passwords, emails, SSNs leaked due to lacking encryption)

Data is Valuable

Sony Pictures, 2014
(passwords, emails, SSNs leaked due to lacking encryption)

TJX Companies Inc., 2006
(94 million credit cards exposed because of weak encryption)

Not everyone who needs to protect data knows how to!

Example:
Securely Store Passwords

Example:
Securely Store Passwords

1. Generate salt & append to password

Example:
Securely Store Passwords

1. Generate salt & append to password
2. Apply a hash function [combined with a random key generator]

Example:
Securely Store Passwords

1. Generate salt & append to password

2. Apply a hash function [combined with a random key generator]

3. Store resulting hash along with salt in user database

Example:
Securely Store Passwords

1. Generate salt & append to password
2. Apply a hash function [combined with a random key generator]
3. Store resulting hash along with salt in user database

Step 1: Generate salt

Do not re-use the salt

Generate random salt for each new user or password change

Do not hard-code the salt

Do not use short salts

```
// Generate a random salt
SecureRandom random = new SecureRandom();
byte[] salt = new byte[SALT_BYTE_SIZE];
random.nextBytes(salt);
```

Step 2: Hashing

Use a secure hash function

Can optionally use a key derivation algorithm

Use a minimum of 1,000 iterations

```
PBEKeySpec spec = new PBEKeySpec(password,
                                 salt,
                                 NUM_OF_ITERATIONS,
                                 LENGTH_OF_HASH);

SecretKeyFactory skf = SecretKeyFactory.getInstance(HASHING_ALGORITHM);
byte[] hashedPassword = skf.generateSecret(spec).getEncoded();
```

Different Levels of “Variability”

- Algorithm(s) to use?
- Sequence and parameters for API calls?

Different Levels of "Variability"

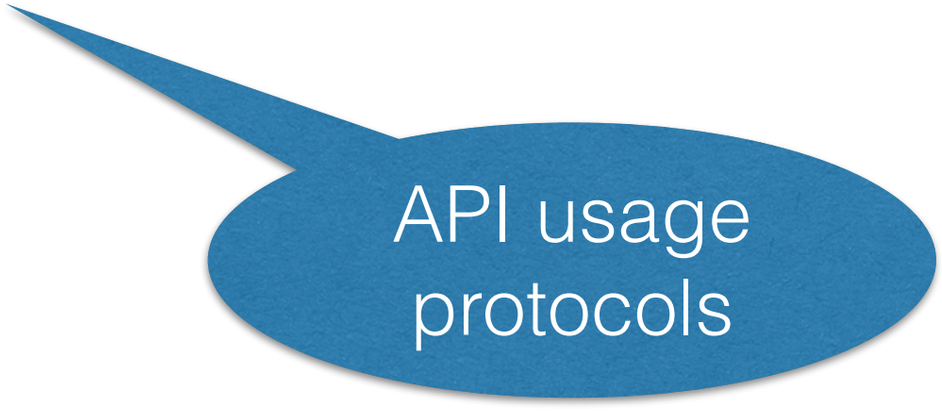
- Algorithm(s) to use?

- Sequence and parameters for API calls?

Different Levels of "Variability"

- Algorithm(s) to use?

- Sequence and parameters for API calls?



Different Levels of "Variability"

- Algorithm(s) to use?

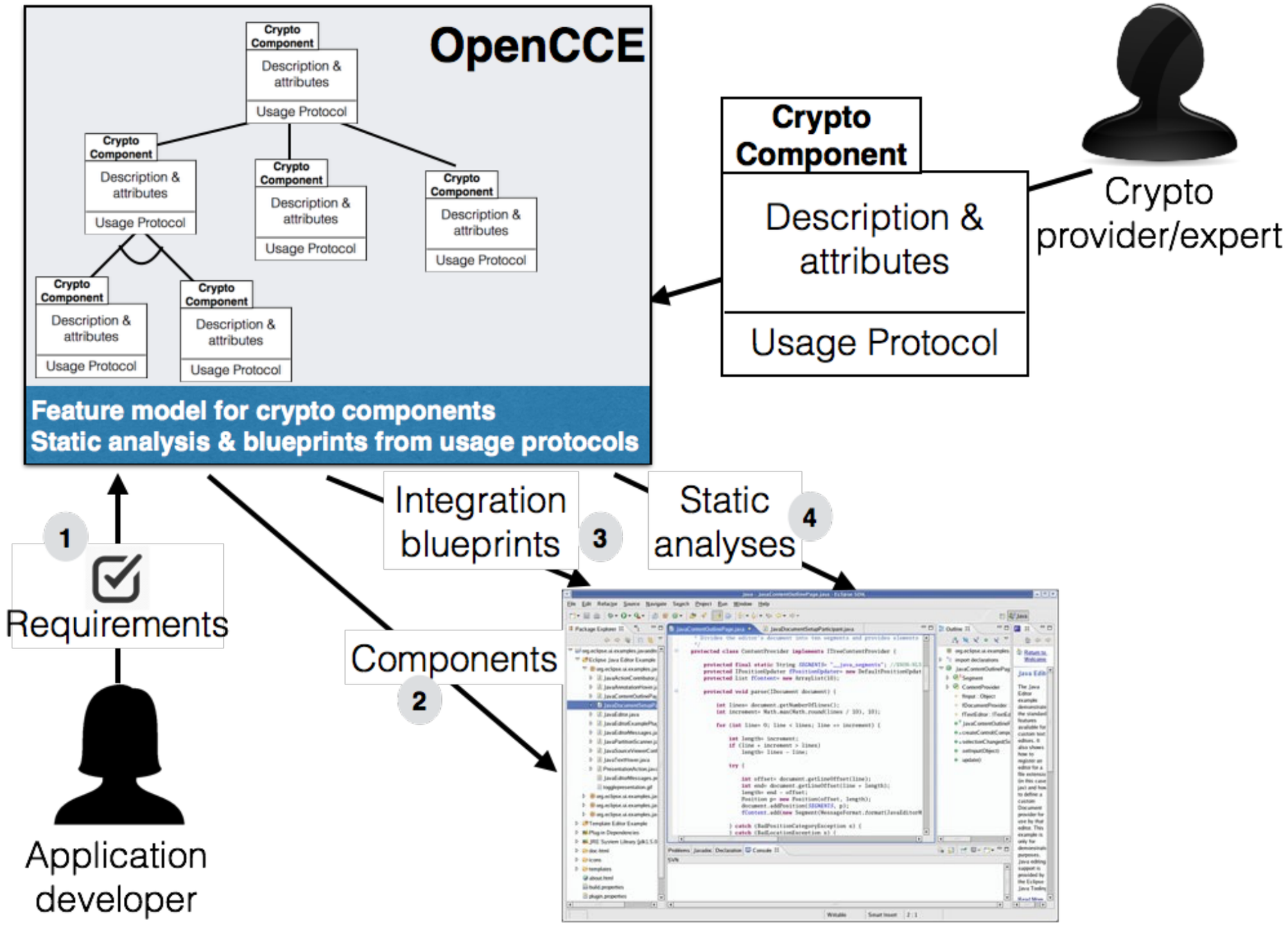
Variability model

Domain knowledge

- Sequence and parameters for API calls?

API usage protocols

Proposed Solution: OpenCCE



Using Clafer for Feature Modeling

- Clafer: class, feature, reference
- General purpose modeling language
- Can be used to define feature models
- Unifies concepts of classes, associations, and properties, and defines a main concept — a clafer.
- Clafers represent properties, types, or references depending on their nesting and syntactic modifiers
- Has an extensive tool suite for instance generation and objective optimization

Password Storing in Clafer

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer
```

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer

abstract KeyDerivAlgm : Algorithm
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer

abstract KeyDerivAlgm : Algorithm
```

```
KeyDerivAlgms
    pbkdf : KeyDerivAlgm
        [name = "PBKDF"]
        [performance = 2]
        [secure]

    bcrypt : KeyDerivAlgm
        [name = "Bcrypt"]
        [performance = 1]
        [secure]
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer

abstract KeyDerivAlgm : Algorithm

abstract Task
    name -> string
```

```
KeyDerivAlgms
    pbkdf : KeyDerivAlgm
        [name = "PBKDF"]
        [performance = 2]
        [secure]

    bcrypt : KeyDerivAlgm
        [name = "Bcrypt"]
        [performance = 1]
        [secure]
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer

abstract KeyDerivAlgm : Algorithm

abstract Task
    name -> string
```

```
KeyDerivAlgms
    pbkdf : KeyDerivAlgm
        [name = "PBKDF"]
        [performance = 2]
        [secure]

    bcrypt : KeyDerivAlgm
        [name = "Bcrypt"]
        [performance = 1]
        [secure]
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```

```
PasswordStoring : Task
    [name = "Password Storing"]
    digestToUse -> Digest ?
    kdaToUse -> KeyDerivAlgm ?
        [kdaToUse = pbkdf => digestToUse]
        [kdaToUse != pbkdf => no digestToUse]
    [kdaToUse || digestToUse]
```

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer

abstract KeyDerivAlgm : Algorithm

abstract Task
    name -> string
```

```
KeyDerivAlgms
    pbkdf : KeyDerivAlgm
        [name = "PBKDF"]
        [performance = 2]
        [secure]

    bcrypt : KeyDerivAlgm
        [name = "Bcrypt"]
        [performance = 1]
        [secure]
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```

```
PasswordStoring : Task
    [name = "Password Storing"]
    digestToUse -> Digest ?
    kdaToUse -> KeyDerivAlgm ?
        [kdaToUse = pbkdf => digestToUse]
        [kdaToUse != pbkdf => no digestToUse]
    [kdaToUse || digestToUse]
```

```
digestToUse = md5
```

```
digestToUse = sha_224
```

```
digestToUse = sha_512
```

```
kdaToUse = bcrypt
```

```
digestToUse = md5
kdaToUse = pbkdf
```

```
digestToUse = sha_224
kdaToUse = pbkdf
```

```
digestToUse = sha_512
kdaToUse = pbkdf
```

Password Storing in Clafer

```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer

abstract KeyDerivAlgm : Algorithm

abstract Task
    name -> string
```

```
KeyDerivAlgms
    pbkdf : KeyDerivAlgm
        [name = "PBKDF"]
        [performance = 2]
        [secure]

    bcrypt : KeyDerivAlgm
        [name = "Bcrypt"]
        [performance = 1]
        [secure]
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```

```
digestToUse = sha_512
```

```
kdaToUse = bcrypt
```

```
PasswordStoring : Task
    [name = "Password Storing"]
    digestToUse -> Digest ?
    kdaToUse -> KeyDerivAlgm ?
        [kdaToUse = pbkdf => digestToUse]
        [kdaToUse != pbkdf => no digestToUse]
    [kdaToUse || digestToUse]
    [no digestToUse.status.insecure]
    [digestToUse => digestToUse.outputSize > 224]
```

```
digestToUse = sha_512
kdaToUse = pbkdf
```

Challenges

Modeling

- Identifying the best boundary between feature model and usage protocol
- Identifying constraints

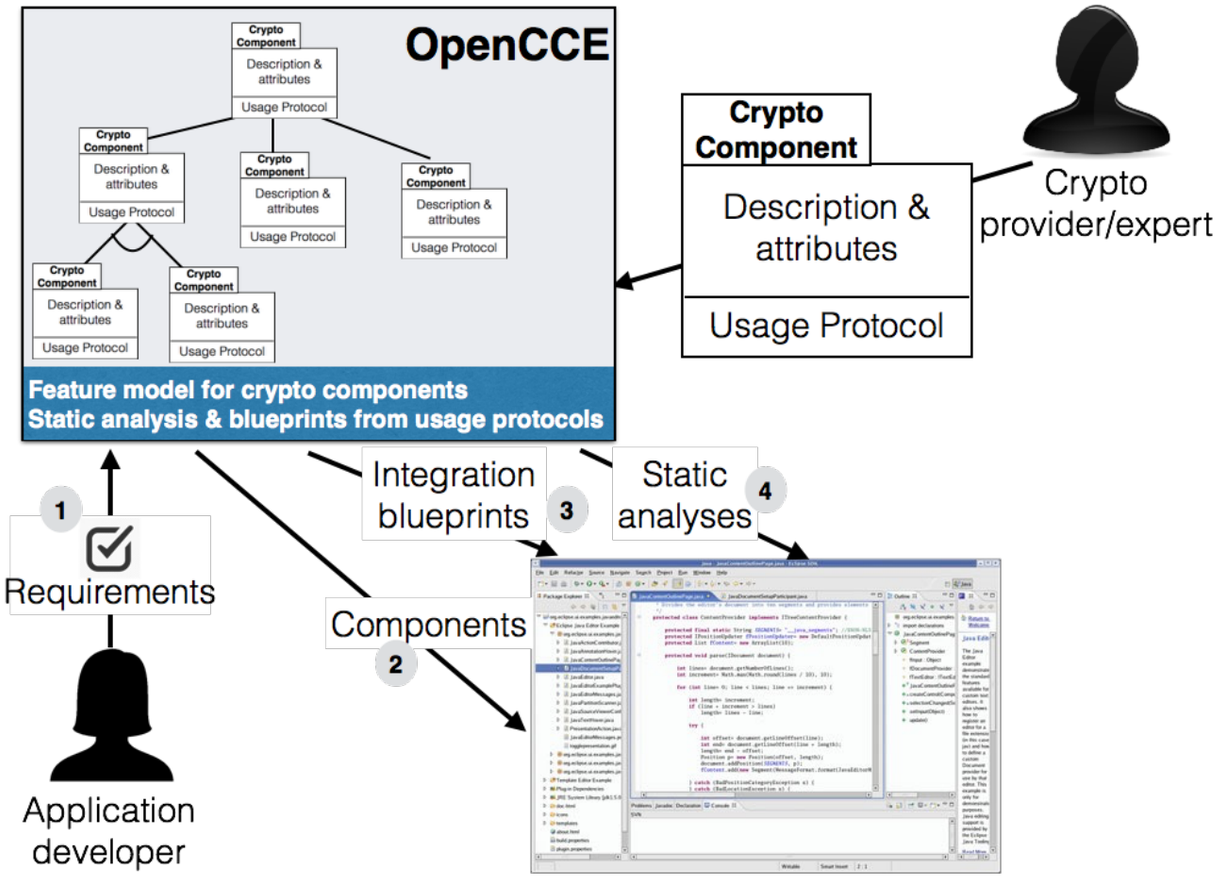
Clafer

- Dealing with redundant information in generated instances
- “Querying” the model
- Configurator design

General Design

- Interfacing between feature model & usage protocol

Variability Modeling of Cryptographic Components




```
abstract Algorithm
    name -> string
    performance -> integer ?
    xor status
        secure
        insecure

abstract Digest : Algorithm
    outputSize -> integer

abstract KeyDerivAlgm : Algorithm

abstract Task
    name -> string
```

```
KeyDerivAlgms
    pbkdf : KeyDerivAlgm
        [name = "PBKDF"]
        [performance = 2
        [secure]

    bcrypt : KeyDerivAlgm
        [name = "Bcrypt"]
        [performance = 1]
        [secure]
```

```
DigestAlgorithms
    md5: Digest
        [name = "MD5"]
        [performance = 4]
        [insecure]
        [outputSize = 128]

    sha_224: Digest
        [name = "SHA-224"]
        [outputSize = 224 ]
        [secure]
        [performance = 2]

    sha_512: Digest
        [name = "SHA-512"]
        [secure]
        [outputSize=512]
        [performance = 3]
```


Sarah Nadi

TU Darmstadt, Germany

sarahnadi.org