Lightning Talk #2

Création d'une bibliothèque unifiée de Cryptanalyse

William Mathey
EPITA - Laboratoire de Recherche de l'EPITA (LRE)
19 mai 2026

Table des matières

01

Rappel

Objectif

Objectif : Développer une bibliothèque unifiée de cryptanalyse.

Sous-objectif : Creer des Graphes pour visualiser les chiffrements (ex : 7eme tour d'AES suivis d'un tour de MIDORI).

Roadmap

Jusqu'à fin mars : Finalisation de l'état de l'art.

Avril - Mai : Recherche itérative de modèles/architectures de représentation.

Juin - Juillet : POC en Python.

Problèmes identifiés dans les approches existantes

- **TAGADA** : manipulation par mots → pas d'opérations booléennes.
- **CLAASP** : manipulation par bits → perd le contexte des mots.
- **BONC** (en développement) : tente de conserver un contexte global, mais encore immature.
- **OCP** : Implémentation trop éloignée des problématiques réelles.
- **Modularité insuffisante** : difficile d'enchaîner des tours de chiffrements différents (ex. AES + Midori + SPECK).

02

Recherche itérative

Première approche : code impératif

Pourquoi

- Semble plus naturel.
- Une fonction pour une action.

Limite rencontrée

- Compliqué à maintenir.
- **Grand manque de modularité.**
- Une approche impérative obligerait presque de la génération sur mesure (automatisée ou non) de chiffrement.

Seconde approche : orientée objet classique

Principes

- Travailler **sur le chiffré globalement**.
- Découpage uniquement si nécessaire (SBox, ShiftColumn).
- Opérations **simultanées** (ex. SBox sur tous les bytes). (Grand gain de complexité)
- **Builders** : un par chiffrement (DES, AES, Midori, Skinny, SPECK, Wrap).
- **Visiteurs** pour l'exécution.

Limite rencontrée

Liaison complexe entre variables et opérateurs. Il faut également penser à toujours vérifier les ensembles de définitions des fonctions et des valeurs.

Approche retenue : tout est variable

Changement de paradigme

- On ne manipule que des **variables**.
- Les **ensembles** sont attachés aux **opérateurs**, pas aux variables.
- Une fonction $f : \mathbb{G}_{2^n} \times \mathbb{G}_{2^n} \to \mathbb{G}_{2^n}$ est représentée par :

$$x \longleftarrow f \longleftrightarrow y$$

- Un opérateur prend n variables en entrée et produit une variable en sortie.
- Chaque appel à une fonction est une **instance différente** (évite les conflits de liaison).

Compléments et Avantages de la nouvelle implémentation

Composants

- **Builders** : un par chiffrement + un builder global.
- **Visiteurs** : exécution récursive (gauche ou droite/gauche).

Avantages

- **Simplicité** : tout est variable, pas de contextes implicites.
- **Modularité** : on chaîne des graphes de chiffrements quelconques.
- **Compatibilité** : symétrique/asymétrique géré uniformément.
- **Graph UML** simplifier.
- **Code** facilement maintenable.
- **Visualisation contrôlée** : on peut sélectionner uniquement certains tours (ex. tours 2 et 3).

UML provisoire

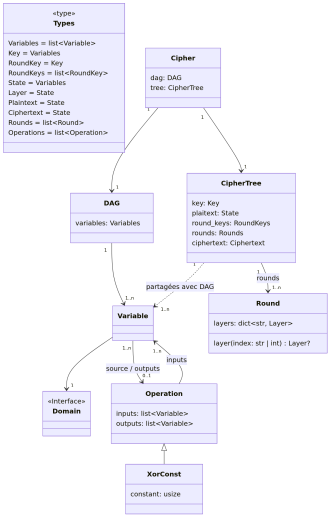

Figure 1 – TAGADA UML

03

Perspectives

Objectif

Roadmap

Juin - Juillet : POC en Python.

Des Questions ?