ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

DaVinciCode

Autocrypton: Cahier des charges

1 Introduction

Le projet autocrypton est inspiré par le projet Crypton par Ashutosh Ahelleya: `https://github.com/ashutosh1206/Crypton`. Ce projet a pour but d'étoffer la liste de Crypton pour lister toutes les attaques fréquentes en CTF dans la catégorie de cryptographie. En deuxième partie, le projet se concentrera sur la création d'un arbre de décision interactif sur une application web.

2 Informations pratiques

2.1 Années recherchées

- A3
- A4
- A5

2.2 Connaissances pré-requises

- Bonnes connaissances et maîtrise de l'arithmétique et de l'algèbre
- Maîtrise de python et/ou Sage
- Maîtrise de l'anglais lu

2.3 Connaissances appréciées

- Maîtrise d'un ou plusieurs domaine de la cryptographie: RSA, AES, ECDSA, LLL, Sbox...
- Participation dans des CTF
- Gitlab CI/CD
- Maîtrise d'un langage web (PHP ou JS) ou framework.

3 Détails techniques

3.1 Arbre de décision

L'interface web de l'arbre de décision sera simplement des cases à cocher, comme par exemple le premier embranchement serait le choix du système cryptographique à attaquer, si l'utilisateur choisit RSA, le deuxième embranchement demandera les valeurs que le challenge donne (N, c, e, phi, d, etc.)

3.2 Liste exhaustives d'attaques

Le premier but du projet est de faire une liste exhaustive des attaques sur les systèmes de cryptographie suivants:

- Block ciphers (AES)
- RSA
- MAC/Hashes
- LLL (Latice reduction)
- DLP (Discrete Logarithm Problem)
- ECDSA (Elliptic Curves)
- Diffie Hellman

Chaque ajout d'attaque doit comprendre les informations suivantes :

- Trouver à l'aide de challenge de CTF: quelle est la faille qui permet cette attaque ?
- Quels sont les paramètres quantifiables permettant cette attaque ?
- Comment peut-on implémenter l'attaque ?

L'implémentation de l'attaque sera à discuter en fonction de sa complexité, si l'implémentation semble avoir besoin de beaucoup de customisation, l'output final de l'arbre de décision sera un snippet python ou sage. A l'inverse, si l'implémentation est très simple et direct, l'arbre de décision pourrait essayer de lancer lui-même l'attaque et l'output final serait le résultat ou flag.