

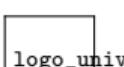
# Operation Rubicon

## Anatomie de la vulnérabilité Minerva

Arnaud Gomes

Université de Bordeaux

February 24, 2026



## La plus grande Supply Chain Attack du 20e siècle

Pendant 50 ans, la **CIA** et le **BND** ont vendu des machines de chiffrement truquées à plus de 100 pays.

- **La Cible** : Crypto AG (Suisse).
- **L'Arme** : Pas de micro, pas de bug logiciel... mais des **Mathématiques**.
- **Le Vecteur** : Hardware "Trusted" (Séries HC-500).



## Le système : Chiffrement par flot (Stream Cipher)

$$c_t = m_t \oplus z_t$$

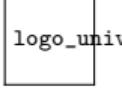
Génération de la suite chiffrante via **LFSR** (Linear Feedback Shift Registers) :

- Polynôme de rétroaction sur  $\mathbb{F}_2[X]$  :

$$f(X) = 1 \oplus c_1 X \oplus \cdots \oplus c_l X^l$$

- **Argument commercial** : Utilisation de polynômes primitifs.
- **Résultat** : Période maximale  $T = 2^l - 1$  (m-suite) et équilibre statistique parfait.

*"Indistinguable du bruit aléatoire pour les tests statistiques."*



**Le problème du LFSR pur :** Vulnérable à une attaque algébrique en  $\mathcal{O}(l^3)$  via Berlekamp-Massey.

**La solution (et la faille) :** Combinaison non-linéaire filtrée.

$$z_t = g(S^{(t)})$$

## La Backdoor Statistique

La fonction  $g$  est truquée pour introduire une **corrélation** vers un registre cible  $L_1$ .

$$P(z_t = L_{1,t}) = 0.5 + \epsilon$$

# L'Attaque : Divide & Conquer

L'espace de recherche s'effondre.

- Complexité théorique (Force Brute) :

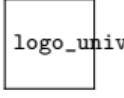
$$\mathcal{O}(2^{L_1+L_2+\dots+L_k}) \approx \infty$$

- Complexité réelle (Minerva) :

$$\mathcal{O}(2^{L_1} + 2^{L_2} + \dots + 2^{L_k}) \approx \text{Quelques secondes}$$

Méthode :

- ➊ Isoler  $L_1$  grâce au biais  $\epsilon$ .
- ➋ Reconstruire  $L_1$ .
- ➌ Soustraire le flux de  $L_1$  et attaquer  $L_2$ .



# Preuve de Concept (Python)

Simulation du biais de corrélation introduit dans la fonction de combinaison :

```
1 def compromised_combiner(l1, l2, l3):
2     # Backdoor: Si l2 & l3 sont nuls, on inverse.
3     # Sinon, on sort l1 tel quel.
4     # Resultat : l1 fuite à 75% (Bias epsilon = 0.25)
5
6     bias_condition = (l2 | l3)
7     if bias_condition == 0:
8         return 1 ^ l1
9     return l1
```

Avec  $\epsilon = 0.25$ , quelques centaines de bits de chiffré suffisent pour converger.

## Hardware is hard... but Math is harder.

- La sécurité par l'obscurité est une dette technique.
- Le matériel fonctionnait "correctement", c'est la spécification qui était hostile.
- **Takeaway** : Vérifiez vos primitives cryptographiques (Kerckhoffs's principle).



Questions ?

