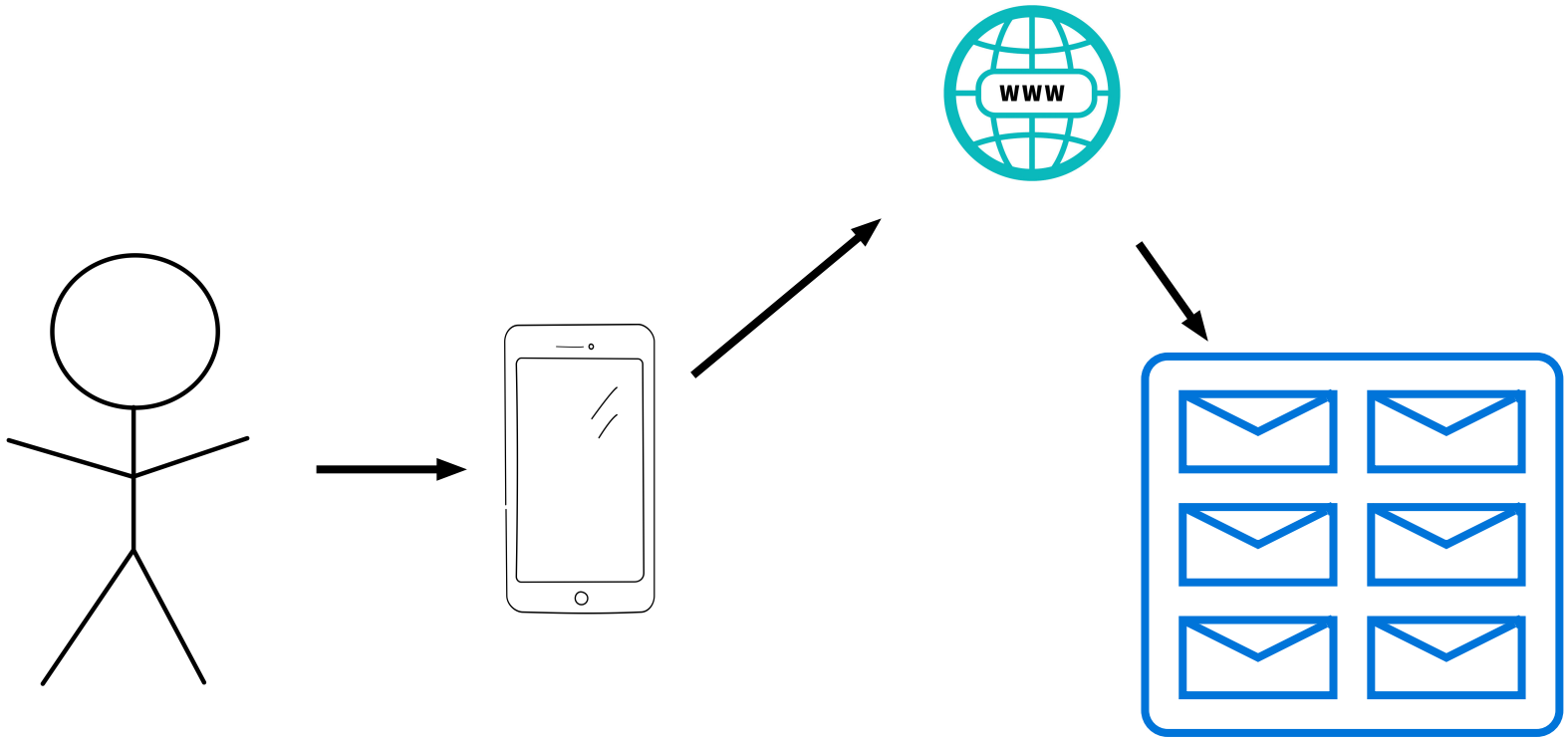
Dolev-Yao-Model-Guided Fuzzing of E-Voting Protocols

Présentation du rapport Bibliographique

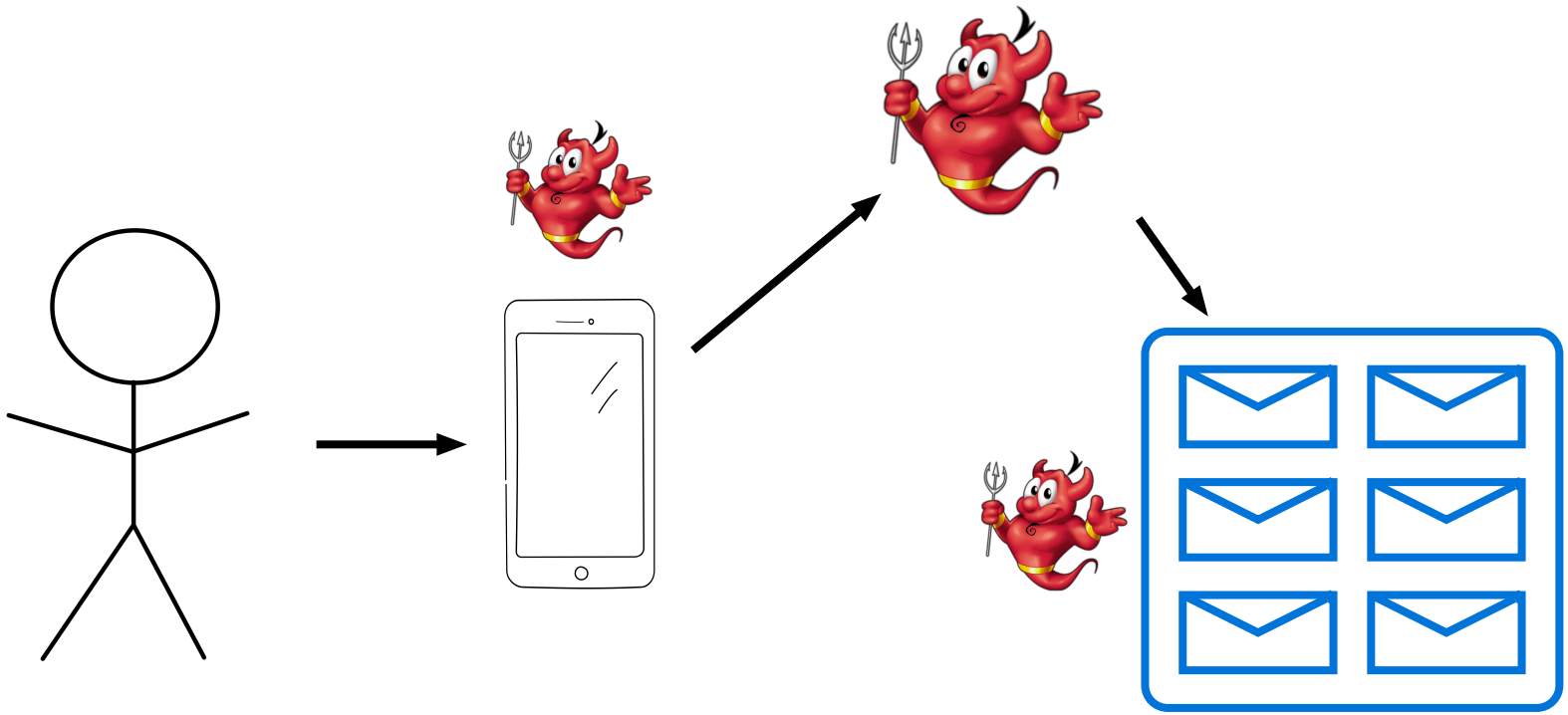
18 février 2026

Benjamin Voisin

Le vote électronique



Le vote électronique



Propriétés de sécurité pour le vote électronique[1]

- Secret du vote

- Confiance dans le résultat

[1] Véronique Cortier et Pierrick Gaudry, *Le Vote Électronique* (Odile Jacob, 2022).

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes

- Confiance dans le résultat

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes

  → Chiffrement asymétrique

- Confiance dans le résultat

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement

→ Chiffrement asymétrique

- Confiance dans le résultat

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

- Confiance dans le résultat

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

→ Bulletin chiffré dissocié de l'identité

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat
  - Éligibilité

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

→ Bulletin chiffré dissocié de l'identité

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat
  - Éligibilité

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

→ Bulletin chiffré dissocié de l'identité

→ Preuve d'identité à côté des bulletins

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat
  - Éligibilité
  - Recorded as Cast

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

→ Bulletin chiffré dissocié de l'identité

→ Preuve d'identité à côté des bulletins

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat
  - Éligibilité
  - Recorded as Cast

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

→ Bulletin chiffré dissocié de l'identité

→ Preuve d'identité à côté des bulletins

→ Moyen de vérifier la présence du vote

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat
  - Éligibilité
  - Recorded as Cast
  - Cast as Intended

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

→ Bulletin chiffré dissocié de l'identité

→ Preuve d'identité à côté des bulletins

→ Moyen de vérifier la présence du vote

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps

→ Chiffrement asymétrique

→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)

→ Bulletin chiffré dissocié de l'identité

- Confiance dans le résultat
  - Éligibilité
  - Recorded as Cast
  - Cast as Intended

→ Preuve d'identité à côté des bulletins

→ Moyen de vérifier la présence du vote

→ Moyen de vérifier le contenu du bulletin

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes
  - Auprès du serveur de vote et des autorités de déchiffrement
  - Durable dans le temps
- Confiance dans le résultat
  - Éligibilité
  - Recorded as Cast
  - Cast as Intended
  - Tallied as Recorded

→ Chiffrement asymétrique
→ Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)
→ Bulletin chiffré dissocié de l'identité

→ Preuve d'identité à côté des bulletins
→ Moyen de vérifier la présence du vote
→ Moyen de vérifier le contenu du bulletin

[1] Cortier et Gaudry, *Le Vote Électronique*.

Propriétés de sécurité pour le vote électronique[1]

- Secret du vote
  - Auprès d'autres personnes → Chiffrement asymétrique
  - Auprès du serveur de vote et des autorités de déchiffrement → Chiffrement homomorphe (on ne déchiffre pas les votes individuellement)
  - Durable dans le temps → Bulletin chiffré dissocié de l'identité
- Confiance dans le résultat
  - Éligibilité → Preuve d'identité à côté des bulletins
  - Recorded as Cast → Moyen de vérifier la présence du vote
  - Cast as Intended → Moyen de vérifier le contenu du bulletin
  - Tallied as Recorded → Preuve que tous les votes sont comptés

[1] Cortier et Gaudry, *Le Vote Électronique*.

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

81eme-1qqfc-be3rx-5t2gk-rznaz

Return codes :

Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

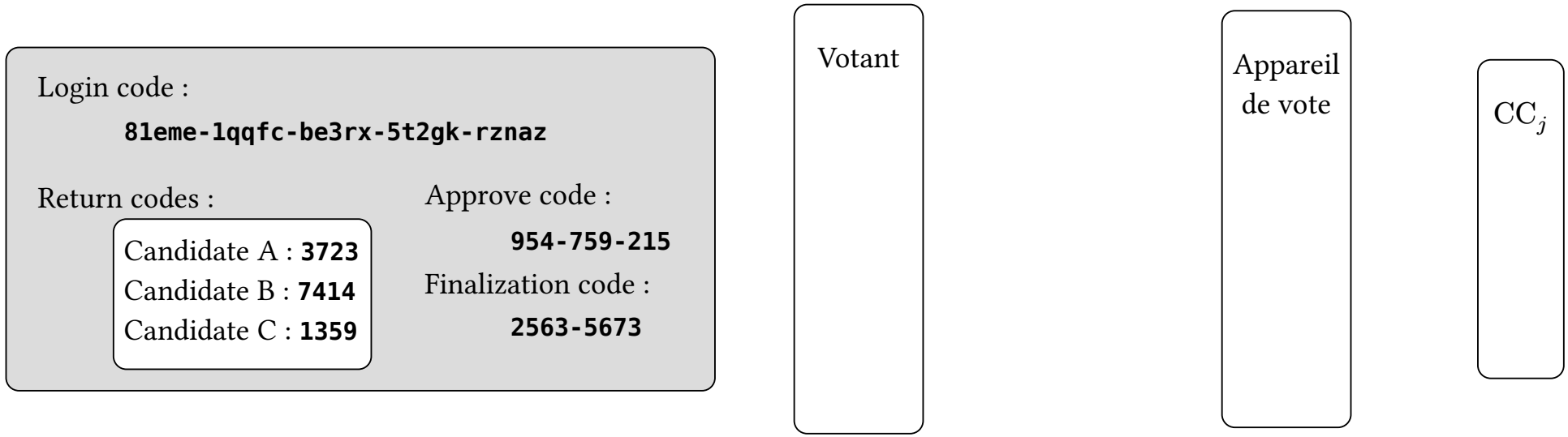
954-759-215

Finalization code :

2563-5673

[1] Véronique Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context », 2025, https://eprint.iacr.org/2025/1625.

Parcours de l'électeur dans le protocole *SwissPost*[1]



[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

`81eme-1qqfc-be3rx-5t2gk-rznaz`

Return codes :

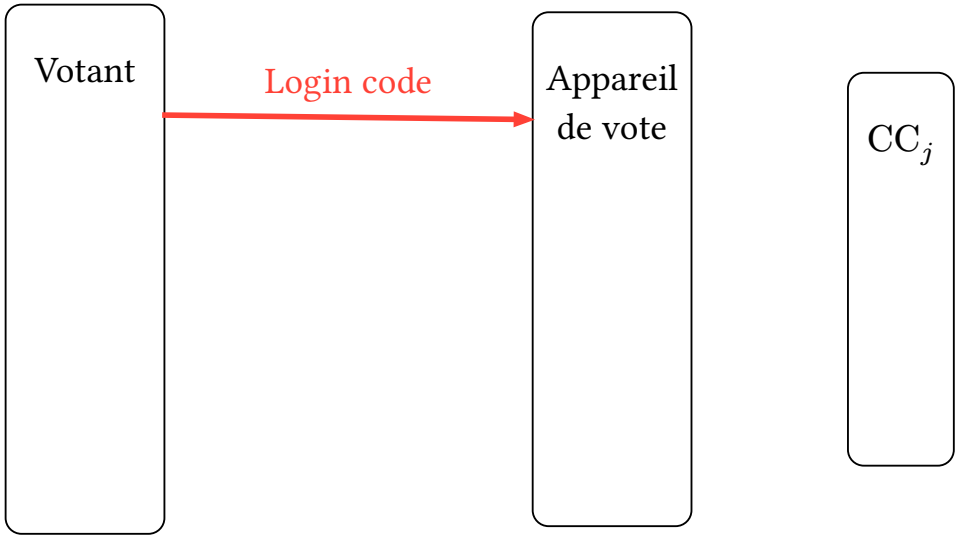
Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

954-759-215

Finalization code :

2563-5673



[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

81eme-1qqfc-be3rx-5t2gk-rznaz

Return codes :

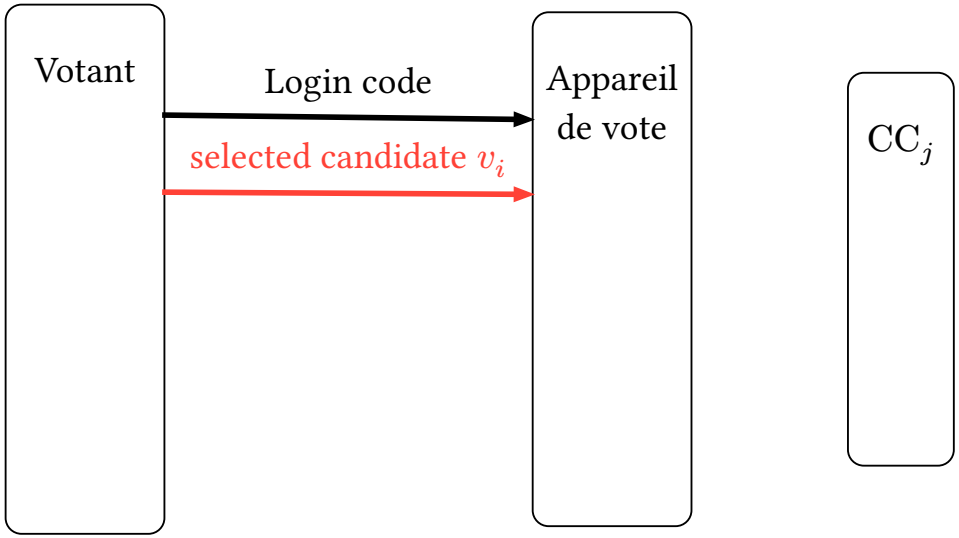
Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

954-759-215

Finalization code :

2563-5673



[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

81eme-1qqfc-be3rx-5t2gk-rznaz

Return codes :

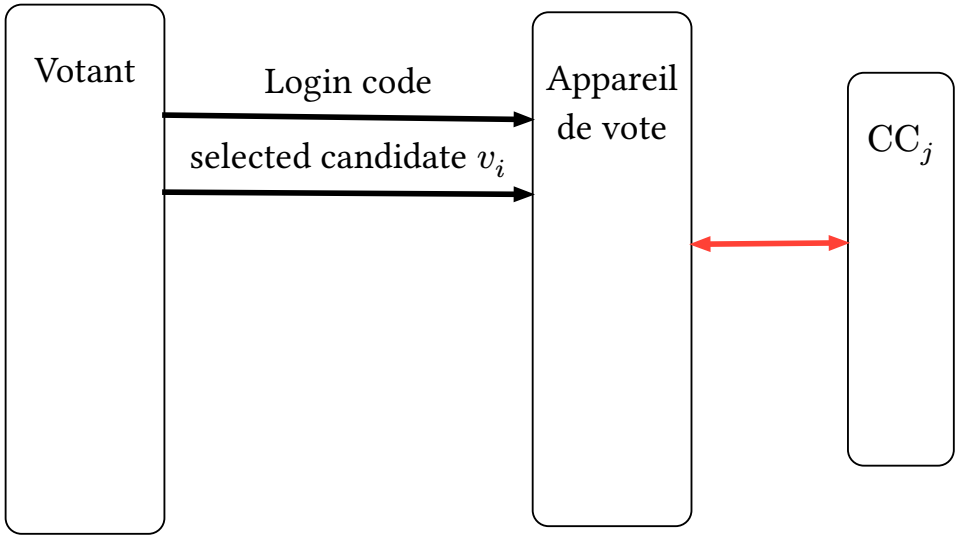
Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

954-759-215

Finalization code :

2563-5673



[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

81eme-1qqfc-be3rx-5t2gk-rznaz

Return codes :

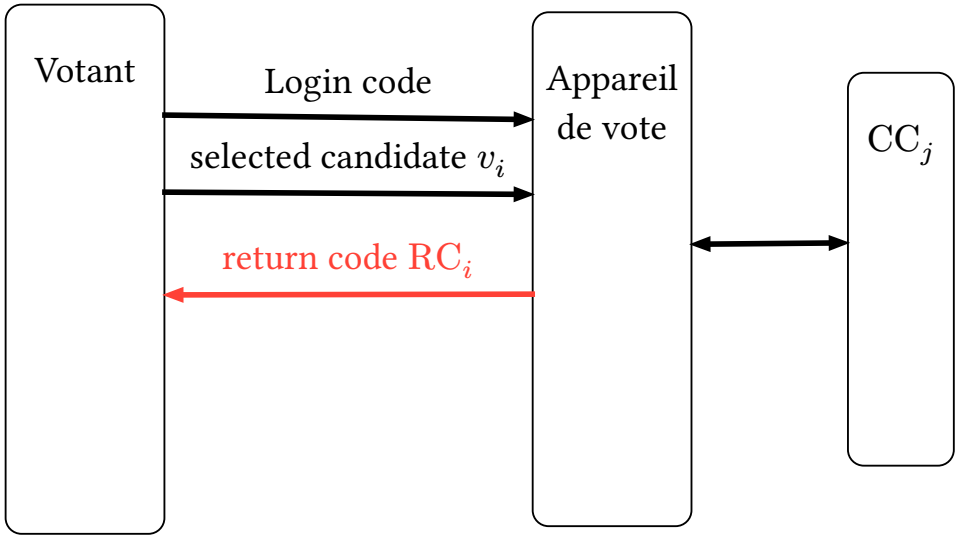
Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

954-759-215

Finalization code :

2563-5673



[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

81eme-1qqfc-be3rx-5t2gk-rznaz

Return codes :

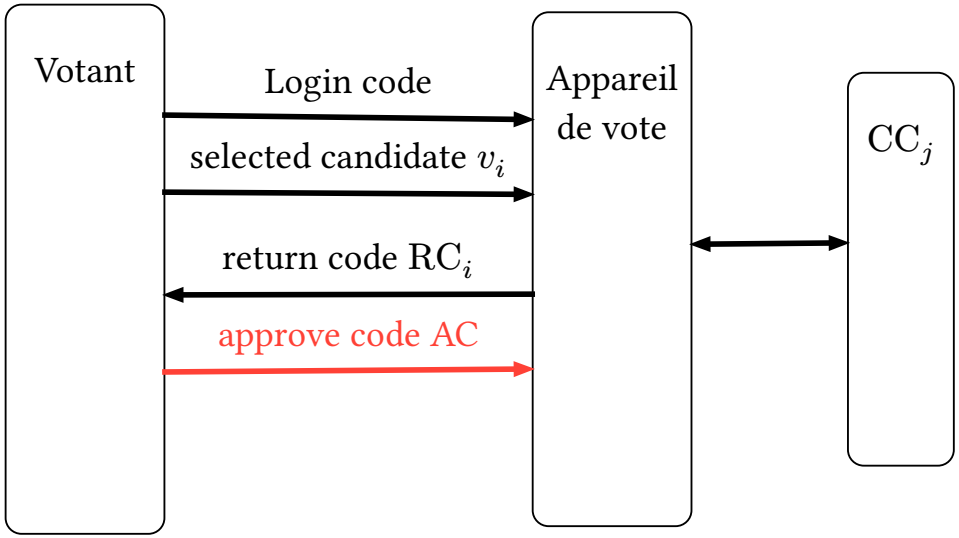
Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

954-759-215

Finalization code :

2563-5673



[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

81eme-1qqfc-be3rx-5t2gk-rznaz

Return codes :

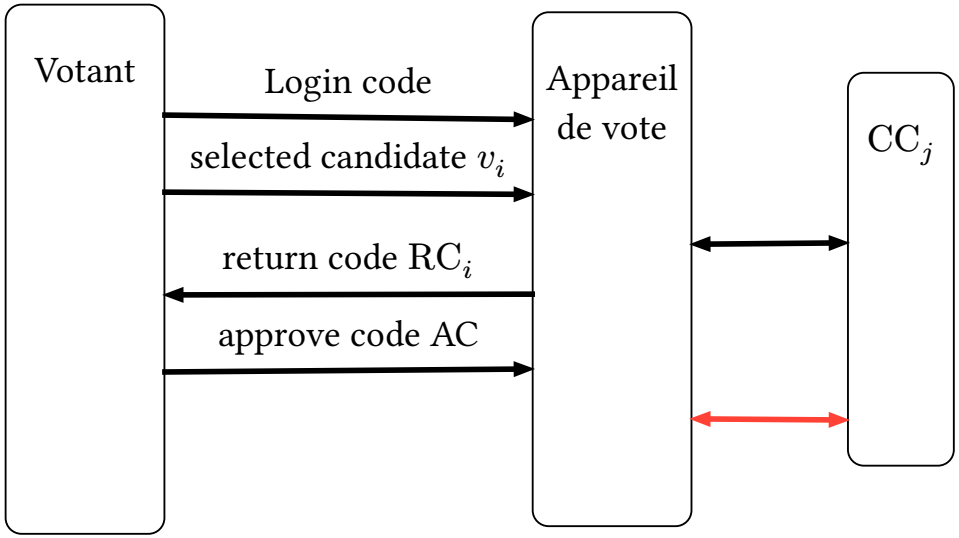
Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

954-759-215

Finalization code :

2563-5673



[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Parcours de l'électeur dans le protocole *SwissPost*[1]

Login code :

81eme-1qqfc-be3rx-5t2gk-rznaz

Return codes :

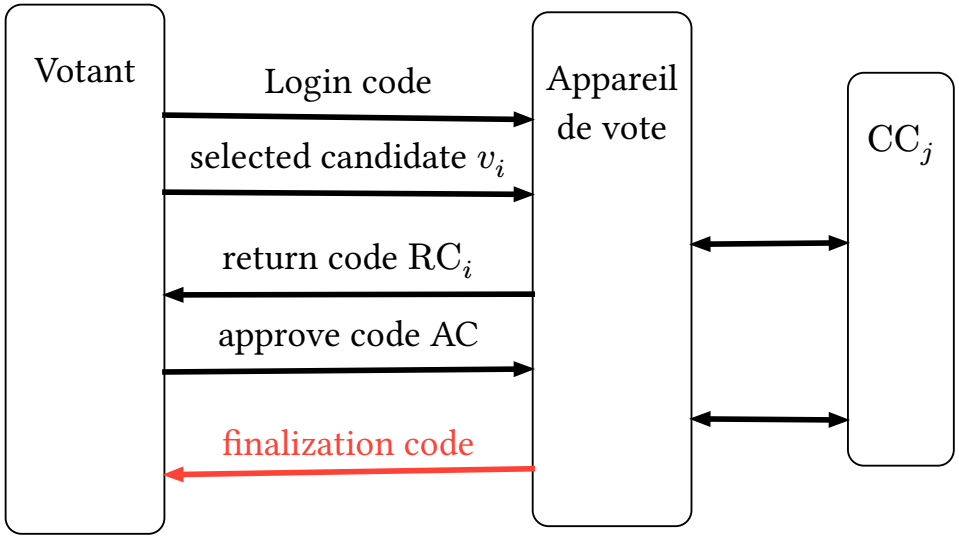
Candidate A : **3723**
Candidate B : **7414**
Candidate C : **1359**

Approve code :

954-759-215

Finalization code :

2563-5673

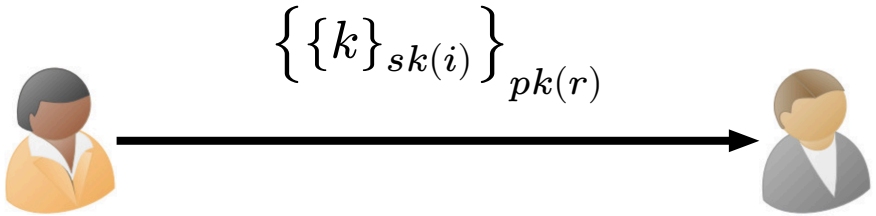


[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Comment s'assurer que ce protocole de vote respecte bien les propriétés de sécurité qu'on souhaite ?

Le modèle de Dolev-Yao[1]

Comment vérifier la sécurité d'un protocole ?

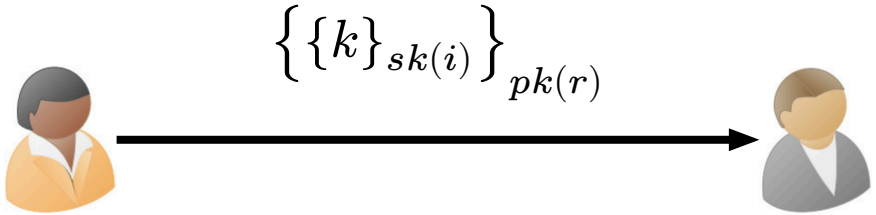


Est-ce que ce protocole (Denning Sacco) est sécurisé ?

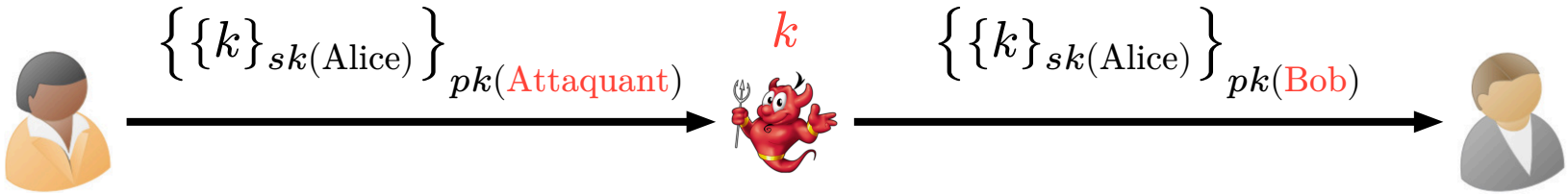
[1]D. Dolev et A. Yao, « On the security of public key protocols », *IEEE Transactions on Information Theory* 29, n° 2 (1983): 198-208, https://doi.org/10.1109/TIT.1983.1056650.

Le modèle de Dolev-Yao[1]

Comment vérifier la sécurité d'un protocole ?



Est-ce que ce protocole (Denning Sacco) est sécurisé ?



[1]Dolev et Yao, « On the security of public key protocols ».

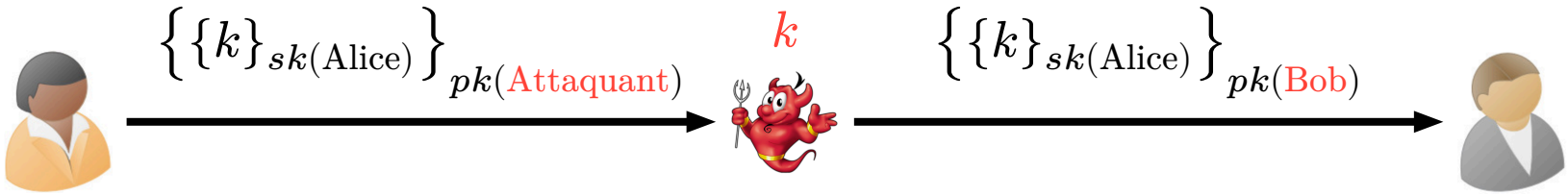
Le modèle de Dolev-Yao[1]

Pour définir un protocole, on donne les fonctions qu'on peut utiliser (e.g. `senc` pour le chiffrement symétrique) ainsi que leur sémantique à l'aide d'une *théorie équationnelle*. Par exemple :

$$sdec(senc(m, k), k) = m$$

Indique qu'on peut déchiffrer un message si on a la clé de chiffrement.

L'attaquant a le contrôle total du réseau



[1]Dolev et Yao, « On the security of public key protocols ».

- Propriétés de trace

- Propriétés d'équivalences

[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

- Propriétés de trace

  Une propriété qui dépend d'évènements au sein du protocole, et de valeur de variables.

- Propriétés d'équivalences

[1]Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

- Propriétés de trace

  Une propriété qui dépend d'évènements au sein du protocole, et de valeur de variables.

Par exemple, pour les propriétés *Cast-as-Intended* et *Recorded-as-Cast* :[1]

```
event(HappyVoter(id, vote)) =>
    BallotRegistred(ballot) & ballot = aenc(pk, vote, random)
```

- Propriétés d'équivalences

[1] Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

- Propriétés de trace

  Une propriété qui dépend d'évènements au sein du protocole, et de valeur de variables.

Par exemple, pour les propriétés *Cast-as-Intended* et *Recorded-as-Cast* :[1]

```
event(HappyVoter(id, vote)) =>
    BallotRegistred(ballot) & ballot = aenc(pk, vote, random)
```

- Propriétés d'équivalences

  Une propriété portant sur *l'indistinguabilité* de deux scénarios d'un protocole.

[1] Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

- Propriétés de trace

  Une propriété qui dépend d'évènements au sein du protocole, et de valeur de variables.

Par exemple, pour les propriétés *Cast-as-Intended* et *Recorded-as-Cast* :[1]

```
event(HappyVoter(id, vote)) =>
    BallotRegistred(ballot) & ballot = aenc(pk, vote, random)
```

- Propriétés d'équivalences

  Une propriété portant sur *l'indistinguabilité* de deux scénarios d'un protocole.

Par exemple, entre 2 choix de votes possible :

```
Voter ( Alice , diff [ ZERO , ONE ]) | Voter (Bob , diff [ONE , ZERO ])
```

[1] Cortier et al., « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ».

Pour prouver ces propriétés, on utilise des outils plus ou moins automatisé :

Proverif[1]

Tamarin[2]

Squirrel[3]

Crytpoverif[4]

---

[1]Bruno Blanchet et al., « Automated Verification of Selected Equivalences for Security Protocols », *Journal of Logic and Algebraic Programming* 75, nº 1 (2008): 3-51.

[2]Simon Meier et al., « The TAMARIN Prover for the Symbolic Analysis of Security Protocols », in *Computer Aided Verification*, *Computer Aided Verification*, éd. par Natasha Sharygina et Helmut Veith (Springer Berlin Heidelberg, 2013).

[3]David Baelde et al., « The Squirrel Prover and its Logic », *ACM SIGLOG News* 11, nº 2 (2024), https://doi.org/10.1145/3665453.3665461.

[4]Bruno Blanchet, « CryptoVerif: A Computationally Sound Mechanized Prover for Cryptographic Protocols », in « Dagstuhl seminar "Formal Protocol Verification Applied" », numéro spécial, *Dagstuhl seminar "Formal Protocol Verification Applied"*, 2007.

Les preuves formelles ne suffisent pas...

- Le modèle symbolique limite les capacités de l'attaquant
- La correspondance entre le modèle formel et le protocole n'est pas assurée[1]
- La correspondance entre le protocole et l'implémentation n'est pas assurée
- L'implémentation peut contenir des bugs[2] :
  - Logiques (e.g., absence de vérifications importantes pour le protocole...)
  - De mémoire (e.g, *buffer overflow*, *use after free*,...)
  - Des crashs dans des chemins rarement accessibles

[1] Véronique Cortier et al., « A privacy attack on the SwissPost e-voting system », 2022.

[2] Thomas Haines et Jarrod Rose, *Vestigial Vulnerabilities in Deployed Verifiable E-Voting Systems*, s. d.

Les preuves formelles ne suffisent pas...

- Le modèle symbolique limite les capacités de l'attaquant
- La correspondance entre le modèle formel et le protocole n'est pas assurée[1]
- La correspondance entre le protocole et l'implémentation n'est pas assurée
- L'implémentation peut contenir des bugs[2] :
  - Logiques (e.g., absence de vérifications importantes pour le protocole...)
  - De mémoire (e.g, *buffer overflow*, *use after free*,...)
  - Des crashs dans des chemins rarement accessibles

Il faut aussi tester l'implémentation !

[1] Cortier et al., « A privacy attack on the SwissPost e-voting system ».

[2] Haines et Rose, *Vestigial Vulnerabilities in Deployed Verifiable E-Voting Systems.*

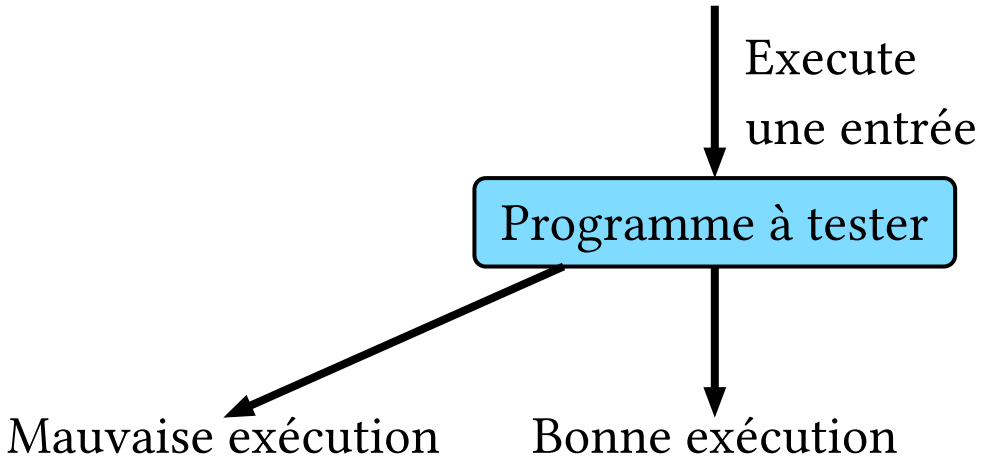
Qu'est-ce que le Fuzzing[1]

[1]Barton P. Miller et al., « An empirical study of the reliability of UNIX utilities », *Commun. ACM* (New York, NY, USA) 33, nº 12 (1990): 32-44, https://doi.org/10.1145/96267.96279.

Qu'est-ce que le Fuzzing[1]

Programme à tester (PUT)

- Instrumenté pour détecter des crashs et bugs de mémoire



[1]Miller et al., « An empirical study of the reliability of UNIX utilities ».

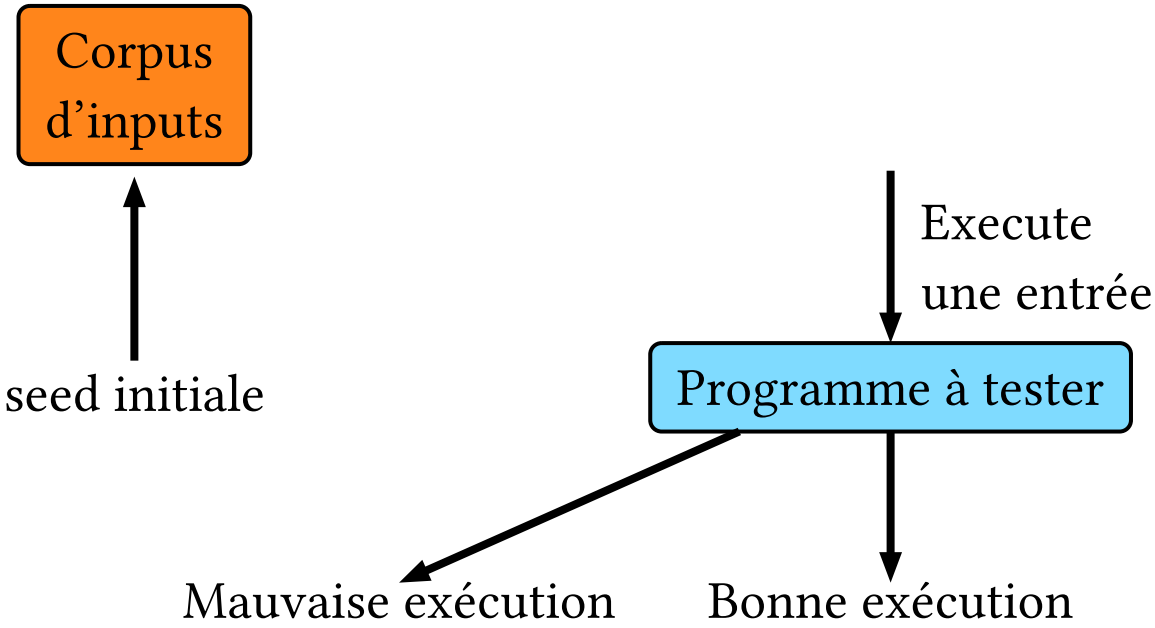
Qu'est-ce que le Fuzzing[1]

Programme à tester (PUT)

- Instrumenté pour détecter des crashs et bugs de mémoire

Corpus d'inputs

- des entrées fonctionnelles ou non



[1]Miller et al., « An empirical study of the reliability of UNIX utilities ».

Qu'est-ce que le Fuzzing[1]

Programme à tester (PUT)

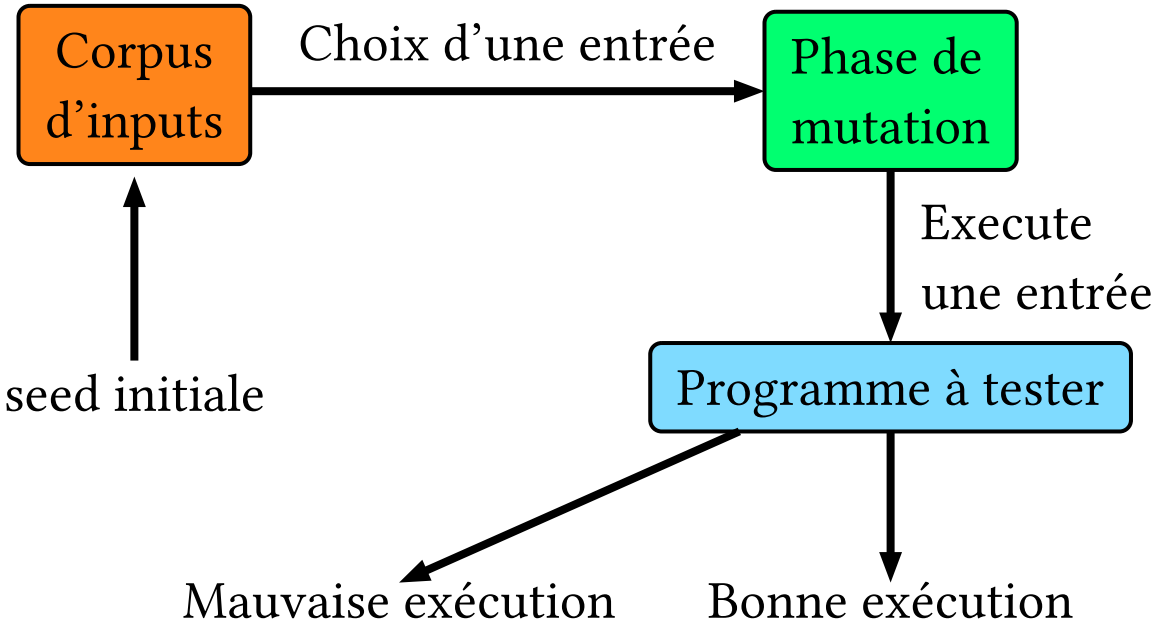
- Instrumenté pour détecter des crashs et bugs de mémoire

Corpus d'inputs

- des entrées fonctionnelles ou non

Phase de mutation

- Modification des entrées



[1]Miller et al., « An empirical study of the reliability of UNIX utilities ».

Qu'est-ce que le Fuzzing[1]

Programme à tester (PUT)

- Instrumenté pour détecter des crashs et bugs de mémoire

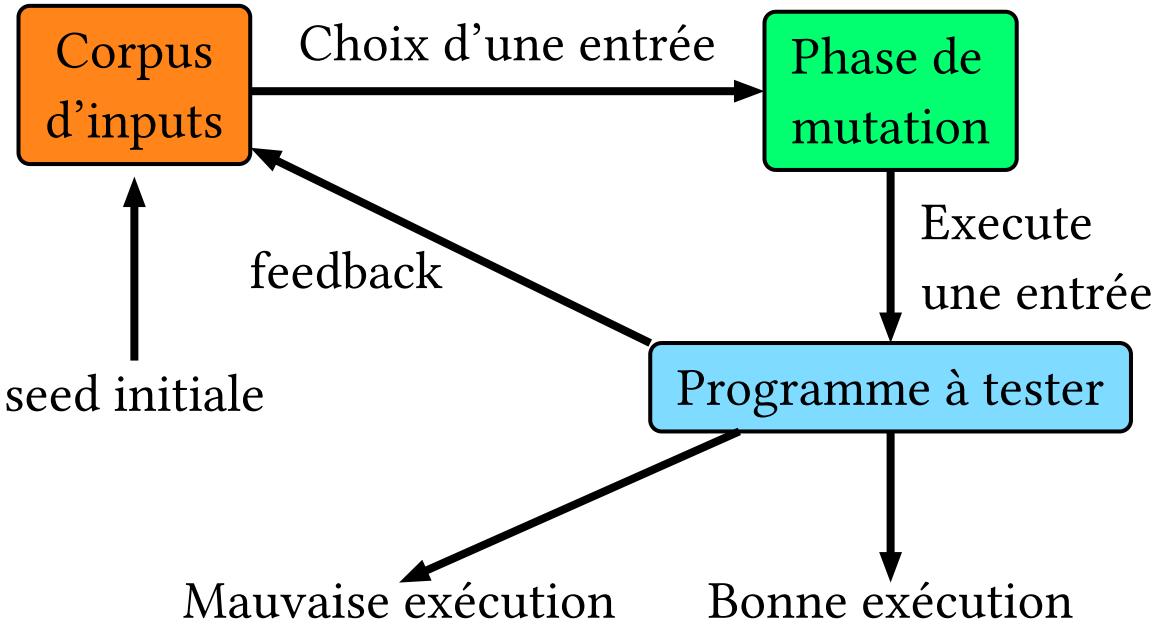
Corpus d'inputs

- des entrées fonctionnelles ou non

Phase de mutation

- Modification des entrées

Une boucle de rétroaction pour conserver les entrées intéressantes



[1]Miller et al., « An empirical study of the reliability of UNIX utilities ».

```
FUZZER(ℂ, t_limit):
 1  𝔹 ← ∅
 2  ℂ ← Preprocess(ℂ)
 3  while t_elapsed < t_limit
 4     conf ← Schedule(ℂ, t_elapsed, t_limit)
 5     test_cases ← InputGen(conf)
 6     // 𝒪_bug is builtin the fuzzer
 7     𝔹′, execinfos ← InputEval(conf, test_cases, 𝒪_bug)
 8     ℂ ← ConfUpdate(ℂ, conf, execinfos)
 9     𝔹 ← 𝔹 ∪ 𝔹′
10  end
11  return(𝔹)
```

Algorithme général de tout fuzzer

Le fuzzing peut avoir d'excellents résultats, et est aujourd'hui devenu une phase incontournable de CI pour tous les logiciels critiques (e.g., OpenSSL).

Cependant,

- Les mutations se font bitwise
  - Très improbable de générer des nouveaux cas tels que « le chiffré du précédent cas »
  - Il est très difficile d'atteindre des états « profonds » du programme, qui nécessitent des branchements logiques improbables
- Permet seulement de détecter des crashs ou des bugs de mémoire, et pas des violations de propriétés de sécurité
- Fonctionne avec 1 programme, ce qui empêche de trouver des problèmes de type « attaquant au milieu »

Il faut faire plus malin si on veut tester des protocoles cryptographiques

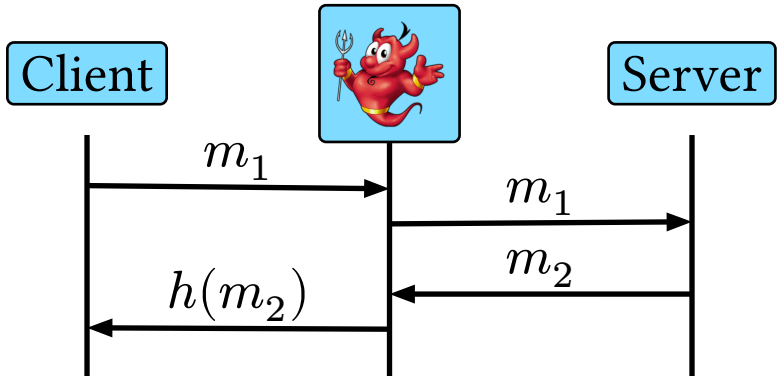
Dolev-Yao-Model-Guided Fuzzing

La solution : Combiner le fuzzing avec le modèle de Dolev-Yao[1] : Le fuzzer manipule des *traces* de protocole.

`out(r,w)` → l'agent `r` émet le message `w`
`in(r, R)` → L'attaquant envoi le message `R` à `r`

Par exemple, cette trace :

```
out(c,m1);in(s,m1);out(s,m2);in(c,h(m2));
```



[1] Max Ammann et al., « DY Fuzzing: Formal Dolev-Yao Models Meet Cryptographic Protocol Fuzz Testing », 2023, https://eprint.iacr.org/2023/057.

Spécificités du DY-fuzzing

Le programme est *instrumenté* pour qu'à chaque exécution, les évènements et les valeurs des variables soient enregistrées, et comparé avec les propriétés de sécurité désirée.

2 éléments spécifiques au DY-fuzzing sont ajoutés :

- Le **mapper**, qui permet de passer d'un terme formel à une suite de bits. Il doit implémenter les fonctions réelle de chiffrement, de déchiffrement etc
- Le **harness**, qui envoi et écoute les messages auprès des programmes à tester, et qui observe l'exécution pour enregistrer les *claims*.

En plus de ça, l'Oracle de bug vérifie les propriétés de sécurité en plus de possible crash et bug de mémoire (Asan).

Cette approche s'est révélée très efficace dans le cas d'implémentations de TLS, avec plusieurs nouvelle CVE découverte[1].

Cependant, elle n'a encore jamais été appliqué à du vote électronique, et plusieurs obstacles empêchent cela :

-
-
-

[1] Ammann et al., « DY Fuzzing: Formal Dolev-Yao Models Meet Cryptographic Protocol Fuzz Testing ».

Cette approche s'est révélée très efficace dans le cas d'implémentations de TLS, avec plusieurs nouvelle CVE découverte[1].

Cependant, elle n'a encore jamais été appliqué à du vote électronique, et plusieurs obstacles empêchent cela :

- Utilisé sur les protocoles simples client-server, là où un protocole de vote contient pleins de composant
-
-

[1] Ammann et al., « DY Fuzzing: Formal Dolev-Yao Models Meet Cryptographic Protocol Fuzz Testing ».

Cette approche s'est révélée très efficace dans le cas d'implémentations de TLS, avec plusieurs nouvelle CVE découverte[1].

Cependant, elle n'a encore jamais été appliqué à du vote électronique, et plusieurs obstacles empêchent cela :

- Utilisé sur les protocoles simples client-server, là où un protocole de vote contient pleins de composant
- Pas de modélisation du modèle de confiance : quel appareil doit fonctionner correctement, lequel peut devenir malveillant
-

[1] Ammann et al., « DY Fuzzing: Formal Dolev-Yao Models Meet Cryptographic Protocol Fuzz Testing ».

Cette approche s'est révélée très efficace dans le cas d'implémentations de TLS, avec plusieurs nouvelle CVE découverte[1].

Cependant, elle n'a encore jamais été appliqué à du vote électronique, et plusieurs obstacles empêchent cela :

- Utilisé sur les protocoles simples client-server, là où un protocole de vote contient pleins de composant
- Pas de modélisation du modèle de confiance : quel appareil doit fonctionner correctement, lequel peut devenir malveillant
- On peut vérifier les propriétés de traces, mais pas les propriétés d'équivalence : pas de vérification de secret du vote

[1] Ammann et al., « DY Fuzzing: Formal Dolev-Yao Models Meet Cryptographic Protocol Fuzz Testing ».

Bibliographie

[1] V. Cortier et P. Gaudry, *Le Vote Électronique*. Odile Jacob, 2022.

[2] V. Cortier, A. Debant, O. Esseiva, P. Gaudry, A. Høgåsen, et C. Spadafora, « A Practical and Fully Distributed E-Voting Protocol for the Swiss Context ». [En ligne]. Disponible sur: https://eprint.iacr.org/2025/1625

[3] D. Dolev et A. Yao, « On the security of public key protocols », *IEEE Transactions on Information Theory*, vol. 29, nº 2, p. 198-208, 1983, doi: 10.1109/TIT.1983.1056650.

[4] B. Blanchet, M. Abadi, et C. Fournet, « Automated Verification of Selected Equivalences for Security Protocols », *Journal of Logic and Algebraic Programming*, vol. 75, nº 1, p. 3-51, 2008.

[5] S. Meier, B. Schmidt, C. Cremers, et D. Basin, « The TAMARIN Prover for the Symbolic Analysis of Security Protocols », in *Computer Aided Verification*, N.

Sharygina et H. Veith, Éd., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, p. 696-701.

[6] D. Baelde, S. Delaune, C. Jacomme, A. Koutsos, et J. Lallemand, « The Squirrel Prover and its Logic », *ACM SIGLOG News*, vol. 11, nº 2, avr. 2024, doi: 10.1145/3665453.3665461.

[7] B. Blanchet, « CryptoVerif: A Computationally Sound Mechanized Prover for Cryptographic Protocols », in *Dagstuhl seminar "Formal Protocol Verification Applied"*, 2007.

[8] V. Cortier, A. Debant, et P. Gaudry, « A privacy attack on the SwissPost e-voting system ». 2022.

[9] T. Haines et J. Rose, « Vestigial Vulnerabilities in Deployed Verifiable E-Voting Systems ».

[10] B. P. Miller, L. Fredriksen, et B. So, « An empirical study of the reliability of UNIX utilities », *Commun. ACM*, vol. 33, nº 12, p. 32-44, déc. 1990, doi: 10.1145/96267.96279.

[11] M. Ammann, L. Hirschi, et S. Kremer, « DY Fuzzing: Formal Dolev-Yao Models Meet Cryptographic Protocol Fuzz Testing ». [En ligne]. Disponible sur: https://eprint.iacr.org/2023/057