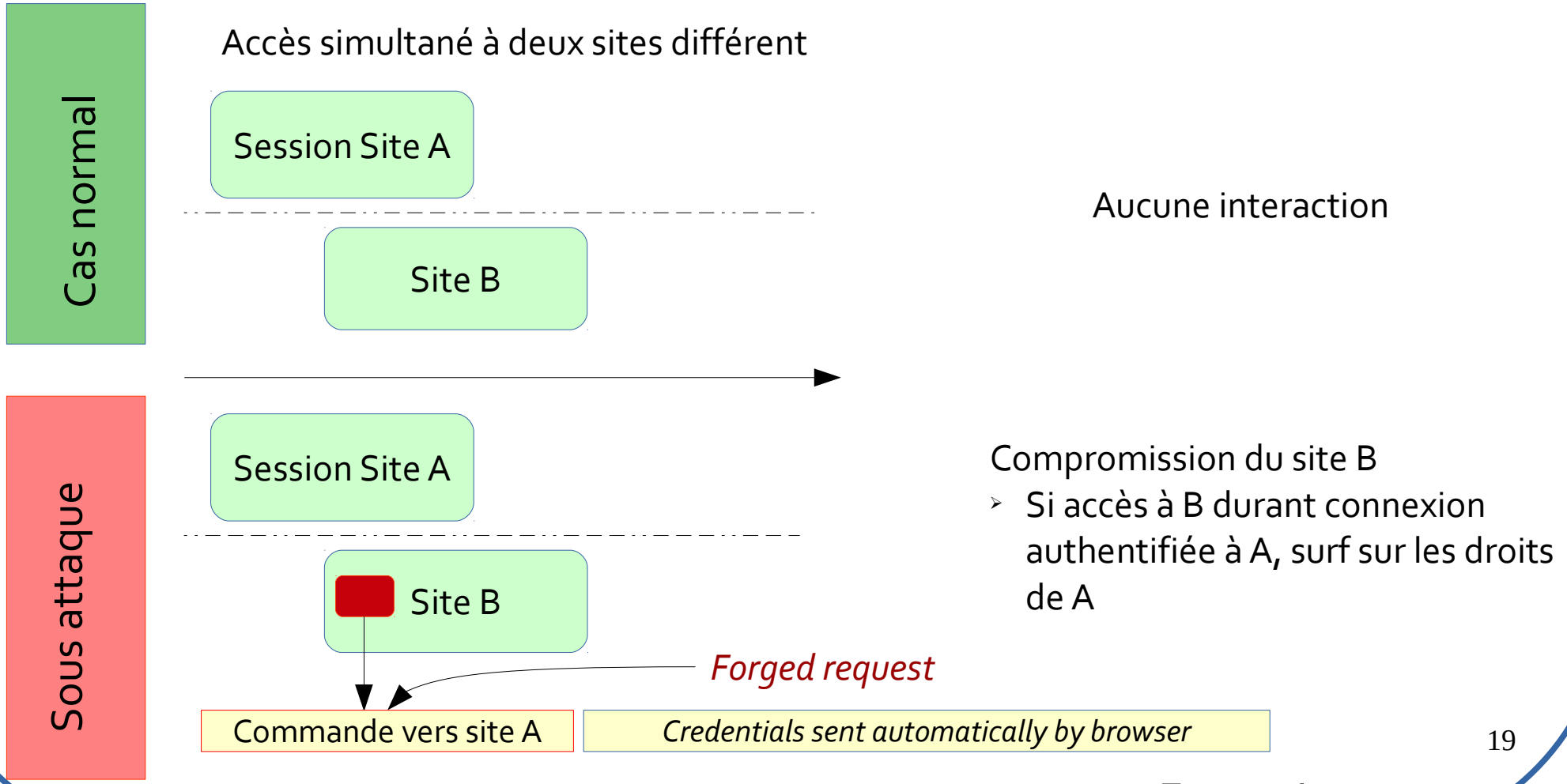
Sécurité des applications

Cross-Site Request Forgery (CSRF)

Utiliser les droits d'un utilisateur pour lui nuire



En pratique...

Sécurité des applications

Cross-Site Request Forgery (CSRF)

Exemple : requête de transaction bancaire vulnérable
Requête normale

http://example.com/app/transferFunds?amount=1500
&destinationAccount=4673243243

Requête du pirate sur un site infecté

<img src="http://example.com/app/transferFunds?
amount=1500&destinationAccount=attackersAcct#“
width="0" height="0" />

L'élément n'a pas besoin d'être visible !

Le pirate ne récupère pas les informations de l'utilisateur !

Sécurité des applications

Cross-Site Request Forgery (CSRF)

Cible	Risques	Conséquences
• Utilisateur final, client du site d'origine	• Exécution de commandes en exploitant les droits d'un utilisateur légitime	• Usurpation d'identité • Destruction de données • Violation de confidentialité

Sécurité des applications

Défenses contre le CSRF

- Authentifications répétées et ergonomie
- Vérification du chemin suivi vers la page demandée
  - Champ HTTP_REFERER de l'en-tête
    - Page de provenance
  - Inclusion d'un jeton aléatoire dans la page ou dans l'URL

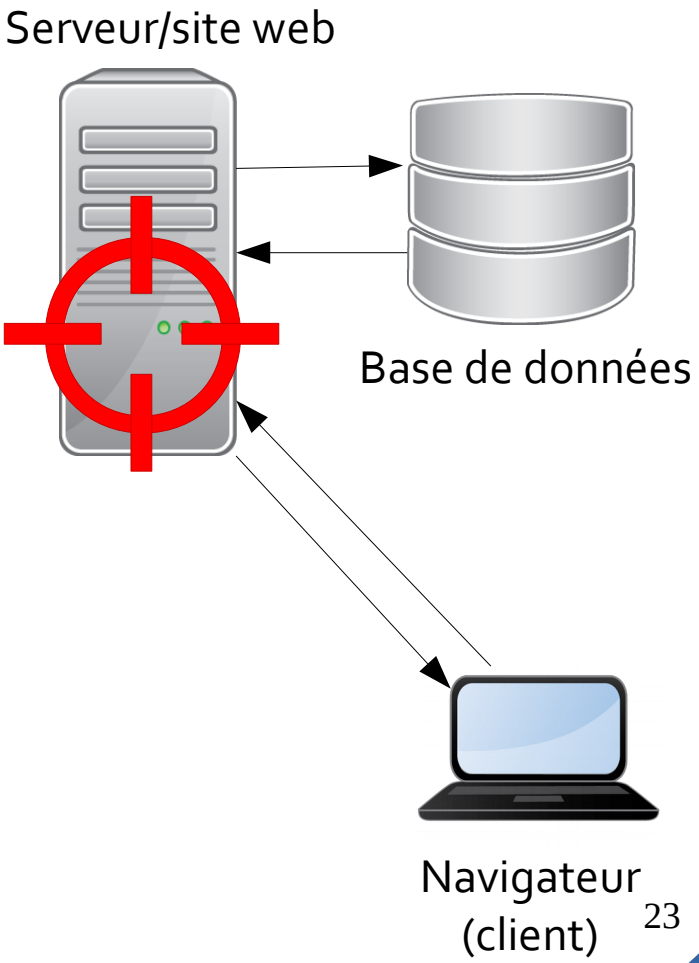
    ✔ Caché dans un champ invisible

    ✔ Dans l'URL…

Sécurité des applications

Dépôt de fichier malicieux

- Propagation / impact
  - Serveur => autres nœuds du réseau
  - Serveur => base de données
  - Serveur => utilisateurs
- Pistes de protection
  - *Upload* vraiment indispensable ??
  - *Upload* si authentification
  - Liste blanche des extensions de fichier autorisées
  - Contrôle du nom de fichier par regex
    - Caractères interdits : « / », « \ », « .. », « $ »…
  - Dépôt dans un répertoire « *non-executable* »
  - Limitation de la taille de fichier (*space* DOS)
  - Scan des fichiers (antivirus)
  - Recommandations propres aux outils utilisés



Practice time !