



## Quatrième Partie



# **Sécurité des Applications**

# Sécurité des applications

---

- Applications Web
  - Manipulation d'URL
  - Attaques cross-site scripting
  - Injection SQL
- XML

# Manipulation d'URL (1/3)

	Protocole	Mot de passe (facultatif)	Nom du serveur	Port (facultatif si 80)	Chemin
exemple 1	http://	user:password@	servername	:80	/dir1/dir2/dir3/index.htm
exemple 2	http://		miageprojet.unice.fr		/twiki/bin/view/Stages/WebHome

- Modification manuelle

- accéder à des pages que lon n'est pas censé voir :

- ✓ ex : Jean Martin accède avec son mot de passe à sa page perso et tente de voir celle de Sophie Durand

- <http://miageprojet.unice.fr/twiki/bin/view/Stages/StageM2Miage2007S1JeanMartin>

- <http://miageprojet.unice.fr/twiki/bin/view/Stages/StageM2Miage2207S1SophieDurand>

- ✓ ex : site web dynamique :

- <http://cible/forum/index.php3?cat=2>

- <http://cible/forum/index.php3?cat=5> (essai d'autre valeurs du paramètre)

- [http://cible/forum/index.php3?cat=\\*\\*\\*\\*\\*](http://cible/forum/index.php3?cat=*****) (provoque une erreur)

—

# Manipulation d'URL (2/3)

---

- Tâtonnement à l'aveugle
  - recherche de répertoire d'administration du site :
    - ✓ <http://cible/admin>
    - ✓ <http://cible/admin/cgi>
  - recherche de fichiers cachés du site distant :
    - ✓ [http://cible/.bash\\_history](http://cible/.bash_history)
  - etc.
- Traversée de répertoire
  - modification de l'arborescence :
    - ✓ <http://miageprojet.unice.fr/twiki/bin/view/Stages/StageM2Miage2007S1JeanMartin>
    - <http://miageprojet.unice.fr/twiki/bin/view/Stages/>
    - <http://miageprojet.unice.fr/twiki/bin/>

# Manipulation d'URL (3/3)

---

- Parades
  - interdire le parcours des pages situées en dessous de la racine du site web (mécanisme de chroot)
  - désactiver l'affichage des fichiers présents dans un répertoire ne contenant pas de fichier d'index
  - supprimer les répertoires et fichiers inutiles
  - empêcher de voir en HTTP les pages accessibles en HTTPS
  - etc.

# • Cross-site scripting (XSS)

---

- Attaques visant des sites qui affichent dynamiquement un contenu fourni par l'utilisateur
  - ex : affichage du nom que l'utilisateur vient de fournir dans un formulaire :  
Le pirate entre du code html ou un script à la place de son nom
- Injection de code dans un site web
  - navigateurs interprètent les scripts du type :
    - ✓ JavaScript, Vbscript, Java, ActiveX, Flash, etc.
  - Balises HTML permettent d'incorporer des scripts :
    - ✓ `<SCRIPT>`, `<OBJECT>`, `<APPLET>`, `<EMBED>`

# Cross-site scripting (XSS)

---

- Conséquences

- le code malicieux a accès aux données partagées par la page Web de l'utilisateur et le serveur (cookies, champ de formulaire)
- le pirate peut vec son code injecté afficher un formulaire demandant les informations d'authentification de l'utilisateur
- le pirate peut rediriger l'utilisateur vers une page qu'il contrôle et qui a la même apparence que le site

# Cross-site scripting (XSS)

---

- Premier type : local
  - faille dans le script d'une page côté client
    - ✓ ex : code JavaScript utilisant le résultat d'une requête d'URL pour écrire du HTML (dans sa propre page) qui sera réinterprété par le navigateur
      - => affichage du html malicieux
    - ✓ possibilité de faire tourner un script malicieux dans la « zone locale » du client

# Cross-site scripting (XSS)

---

- Second type : non permanent
  - le plus commun
  - données fournies par le client Web utilisées telles quelles par les scripts du serveur pour fournir une page de résultats
- Troisième type : permanent
  - très dangereux
  - données fournies par l'utilisateur stockées sur un serveur, puis réaffichées sans encodage préalable des caractères spéciaux HTML
  - ex : forum où les utilisateurs peuvent poster des textes avec des balises HTML

# Cross-site scripting (XSS)

---

- Parades
  - sanitisation (*sanitation*) des données
    - ✓ vérifier le format des données entrées par les utilisateurs
    - ✓ retraiter le HTML de sortie via le CMS (**C**ontent **M**anagement **S**ystem)
    - ✓ filtrer les variables affichées ou enregistrées avec des caractères '<' et '>'
    - ✓ utiliser `<noscript>.....</noscript>`

# Injection SQL

---

- Injection de code de type SQL dans les champs d'un formulaire ou dans les variables d'une page web pour provoquer leur exécution.
  - des formulaires non traités peuvent lancer leurs propre requêtes SQL

- Exemple :

```
SELECT * FROM utilisateurs WHERE nom= "$nom";
```

Formulaire => \$nom ≡ **"toto" OR 1=1 OR nom="titi"**

```
SELECT * FROM utilisateurs WHERE nom="toto" OR 1=1 OR nom="titi";
```

=> clause WHERE toujours satisfaite

=> retourne les enregistrements de tous les utilisateurs

# Injection SQL

- Exemple :

Site web dynamique (en PHP) utilisant une connexion des utilisateurs par 'logon/mot de passe' stockés dans une base SQL.

```
SELECT uid WHERE name= '(nom)' AND password ='(mot de passe chiffré)'
```

Formulaire => Utilisateur : Martin

Mot de passe : secret

```
SELECT uid WHERE name= 'Martin' AND password ='17C3a...8b91D'
```

Formulaire => Utilisateur : Martin' --

Mot de passe : bidon

```
SELECT uid WHERE name= 'Martin' -- ' AND password ='ba137...95f37'
```

**ce qui suit – est un commentaire en PHP**

```
=> SELECT uid WHERE name= 'Martin'
```

L'attaquant peut donc se connecter en utilisant le logon de Martin et n'importe quel mot de passe

# Injection SQL

---

- Parades

- ✓ vérifier le format des données entrées par les utilisateurs
- ✓ ne pas afficher de messages d'erreurs affichant la requête SQL en erreur
- ✓ supprimer les comptes non utilisés
- ✓ utiliser des comptes avec mot de passe
- ✓ restreindre au strict nécessaire les privilèges des comptes utilisés

# Sécurité XML

---

- Cryptage XML

- W3C Recommendation 10 December 2002
- Objectif : permettre le cryptage de (parties de) données XML échangées sur le Web
  - ✓ Un processus pour crypter et décrypter les contenus
  - ✓ Une syntaxe XML pour représenter
    - ◆ les contenus cryptés
    - ◆ les informations permettant le décryptage par le destinataire concerné
- <http://www.w3.org/TR/xmlenc-core/>

# Sécurité XML

---

- Signature XML
  - W3C Recommendation 10 June 2008
  - Objectif : assurer l'intégrité et la non répudiation des données ainsi que l'authentification de l'auteur
    - ✓ Une syntaxe XML pour représenter la signature des ressources Web
    - ✓ Des procédures pour calculer ces signatures (clé privée)
    - ✓ Des procédures pour les vérifier (clé publique)
  - <http://www.w3.org/TR/xmlsig-core/>

# Plan général

---

- Introduction
- Principes de Bases de la Sécurité de l'Information
- Cryptographie
- Sécurité des Réseaux
- Sécurité des Applications
- Politique de sécurité
- Conclusion