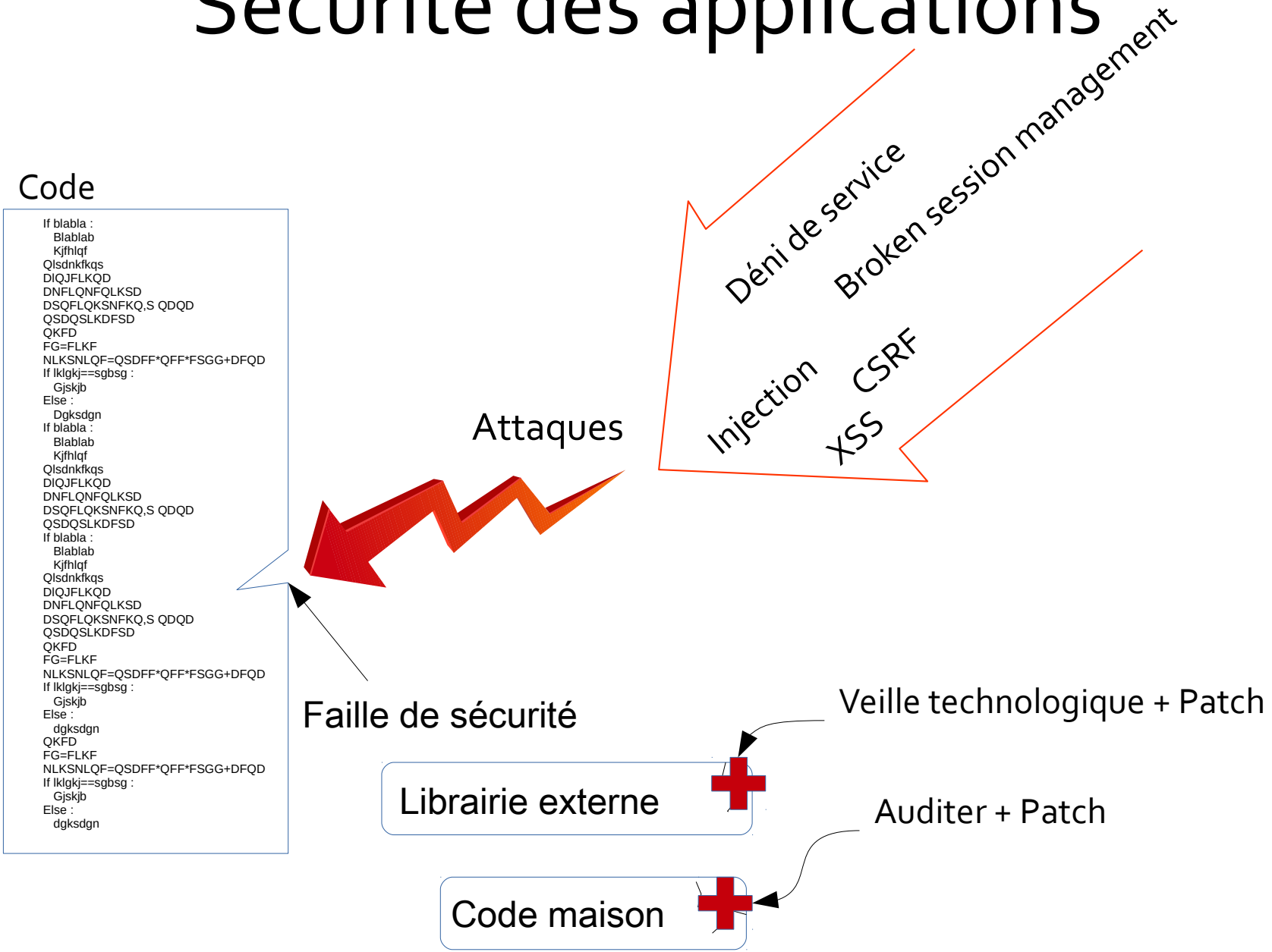
Sommaire

- Généralités
- Protéger la salle info
- Protéger le réseau interne
- Protéger les accès distants
- Protéger les postes de travail
- Le *Social Engineering* et le MICE
- La sécurité des applications
- L'authentification
- La veille technologique
- Conclusion

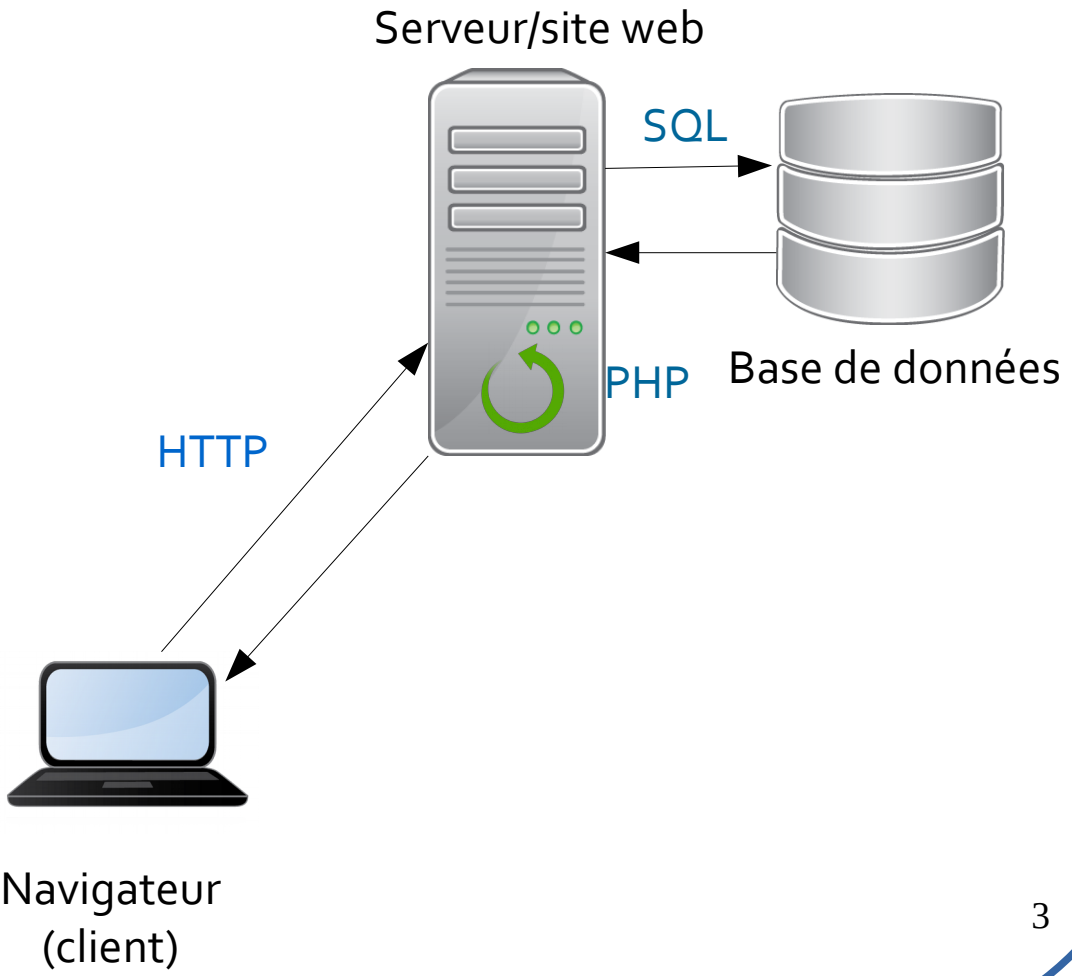
Sécurité des applications



Sécurité des applications

Pourquoi un focus sur le web ?

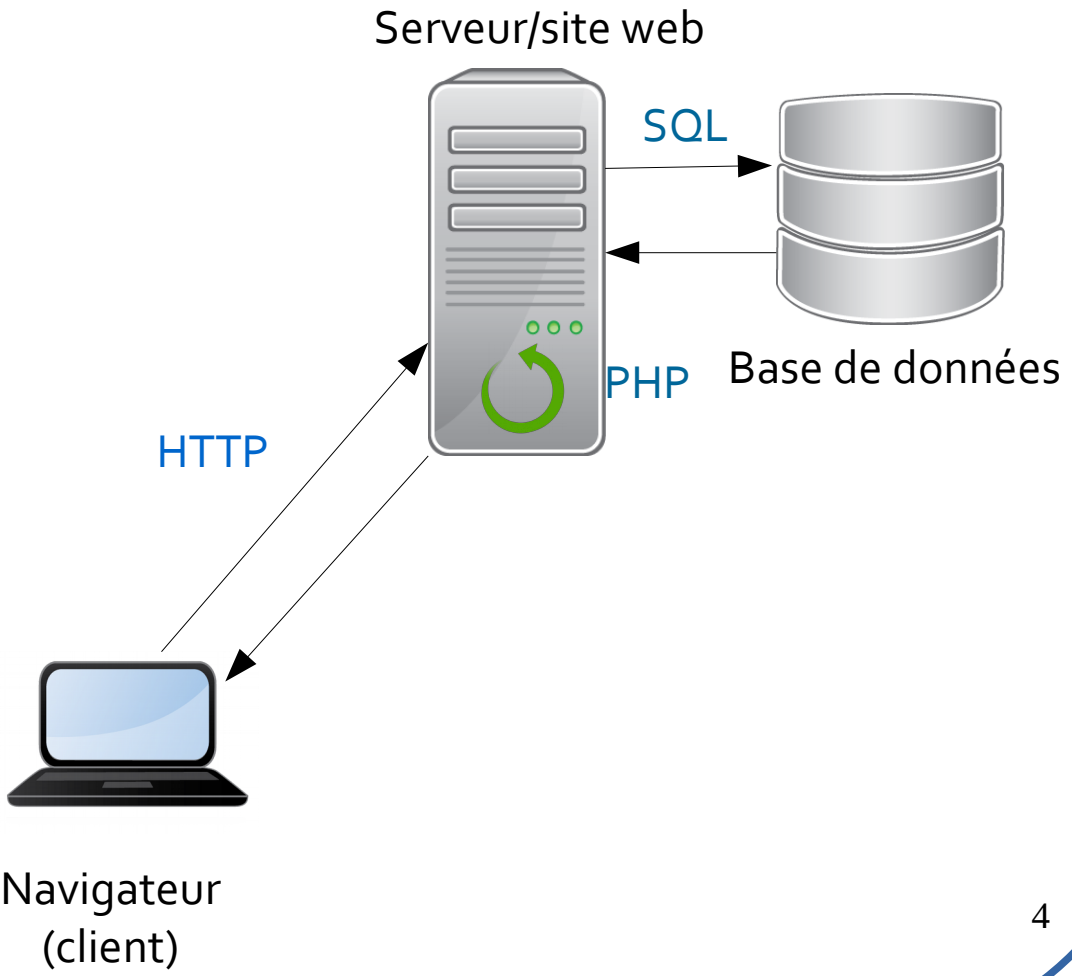
- Applications web
  - Du HTML/CSS au PHP/MySQL
    - Statique vs dynamique
    - Services innovants
    - Accessibles via un client léger, quel que soit le terminal
      - Adoption
  - Concurrence
    - Prudence et sécurité
  - Accès à des données sensibles
    - Divers aspects de la vie quotidienne
      - Site web des services de l'État
      - Finance
      - Travail
    - Attractivité pour les pirates



Sécurité des applications

Pourquoi un focus sur le web ?

- Risques à voir
  - Injection
  - *Binary Planting*
  - XSS
  - CSRF
  - …



Sécurité des applications

Injection

Cas normal

Contribution utilisateur

Base connue → Technologie + Langage → Résultat dynamique désiré

Sous attaque

Contribution pirate

Base connue → Technologie + Langage → Résultat indésirable

En pratique ?

Sécurité des applications

Command injection

Une application passe du contenu utilisateur à un *system shell*

Exemple : Wrapper de 'cat'

```
int main(char* argc, char** argv) {
                char cmd[CMD_MAX] = "/usr/bin/cat ";
                strcat(cmd, argv[1]);
                system(cmd);
        }
```

./mycat monbeausapin.txt

./mycat ; rm -rf

Sécurité des applications

SQL injection 1/2

Les données sont utilisées pour personnaliser une requête SQL

Exemple : Vérifier la disponibilité du produit dont l'ID a été passé dans le champ 'product'

```
Select count(*) from marchandise where
marchandise_id = ‘product’ ;
```

7245362

1’; drop table users #

Fin de requête forcée !

Sécurité des applications

SQL injection 2/2

Les données sont utilisées pour personnaliser une requête SQL

Exemple : modèle basique de récupération d'un dossier médical

```
SELECT * FROM dossier
    WHERE patient = ‘patient_name’
    AND sscode = ‘ss_number’;
```

```
SELECT * FROM dossier    WHERE patient = 'Louis'
                                AND sscode = ‘45225523434522’ ;
```

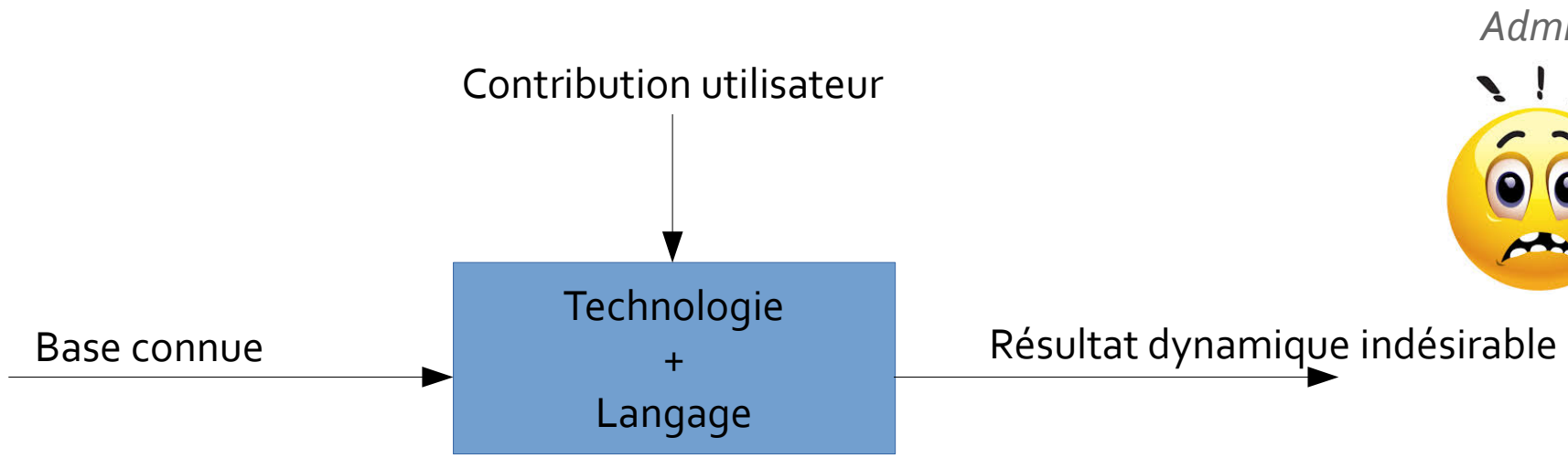
```
SELECT * FROM dossier    WHERE patient = 'Louis'
                                AND sscode = ‘2’ OR ‘1’=’1’ ;
```

Sécurité des applications

	Cibles	Risques	Conséquences
SQL Injection	Base de données contenant des informations sensibles	Violation de • Confidentialité : accès frauduleux • Intégrité : modification des données • Authenticité : récupération et utilisation de données d'accès	• Perte de réputation • Pertes financières • Retard par rapport à la concurrence • Faillite
Command injection	Système de fichiers Programmes installés Gestion des utilisateurs …	Violation de • Confidentialité : récupération de fichiers, de frappes… • Intégrité : création de compte, assignation de droits, installation de programmes malveillants • …	

Sécurité des applications

Défense contre l'injection

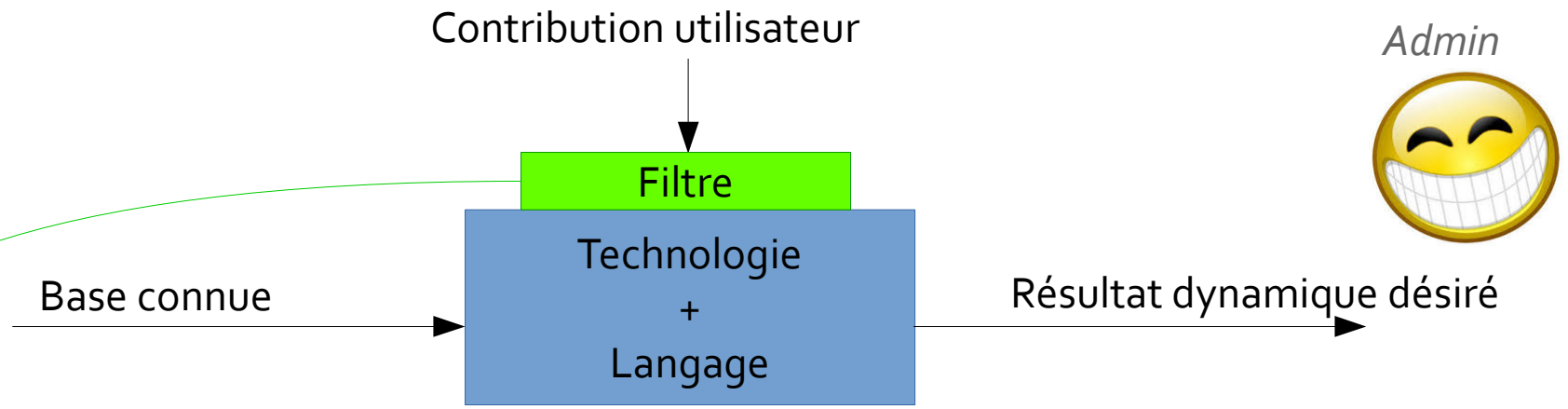


L'utilisateur entre des chaînes de caractères interprétables

- Ont du sens dans le langage associé à la technologie réceptrice

Sécurité des applications

Défense contre l'injection



Désinfection des données utilisateur

- *Contrôle de taille des données entrées*
  - ✔ *Prévention de buffer overflow…*
- *Special character escaping*
  - ✔ Transformer les caractères spéciaux en leur représentation littérale
- Syntax error management
  - ✔ Ne pas donner d'indices sur le nom des tables et l'organisation de l'infrastructure

Sécurité des applications

Défense contre l'injection SQL

1) Utiliser des *prepared statements* / requêtes paramétrées
   - ✔ Distinction claire entre code SQL à interpréter et données non-interprétées
2) Vérification des entrées
   - ✔ Dictionnaire de valeurs et liste blanche
   - ✔ Transformation du texte en valeur numérique/booléenne…
3) *Special character escaping*
   - ✔ Remplacement des caractères spéciaux
   - ✔ Dépendance à la base de données : utiliser mysql_real_escape_string pour MySQL, par exemple

Et en général, minimiser la quantité d'informations retournées à l'utilisateur en cas d'erreur !

It's ~~morphing~~ practice time !