



# PRÉSENTATION DES DIFFÉRENTES VULNÉRABILITÉS WEB

2017



## 1. Client-Side

1. Cross-Site Scripting
2. Cross-Site Request Forgery

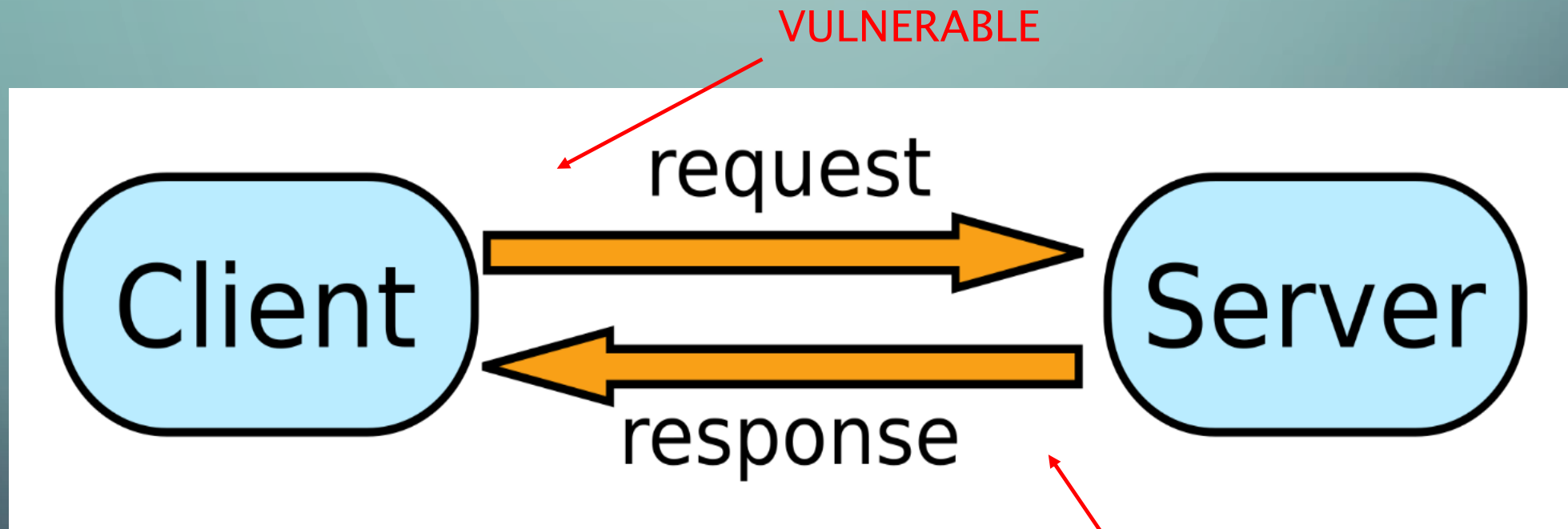
## 2. Server-Side

1. SQL Injection
2. Remote Code Execution
3. Privilege Escalation
4. File Inclusion
5. Full Path Disclosure
6. Server Misconfiguration
7. PHP Type Juggling

## 3. Misc

1. Google Dorks
2. Bug Bounty Programs

# CLIENT SIDE



# CROSS-SITE SCRIPTING

Compte rendu AG Pédago 16/12/16 **Pedagogy** **Representative** **General**

IOS Swift  
OCaml  
Ruby On Rails (in beta test in December, will be available for the January session)

**forum.intra.42.fr indique :**  
mdr cc


OK


Results: between 100 and 150 people did not reach the lvl 5 in the time allotted and that total 226 people were combined. Many students have put the rating school at a time of their curriculum for various reasons and have

Reminder of OI concerning BIH:  
The BHs are benchmarks set up by the pedago to locate your advance in the curriculum. We must avoid telling "retry" and find myself trap when you arrive at the deadline.

The failure of a BH must lead to a questioning on the part of the student. If there is a problem, please do not learn about the organization of work, exchange on your methods.

The BHs are not the one to tackle the students slower but to identify those who do not invest in their school

 **wliu**  
Warren, thank you for translating. One thing to add in translation of piscine part:  
Access from lvl 5: PHP.  
Access post stage: (see message above)

 **prichard**

(MESSAGE PREVIEW)

`<script>alert('mdr cc')</script>`



# DESCRIPTION DE LA VULNÉRABILITÉ

Une vulnérabilité de type Cross Site Scripting ( Communément appelée XSS ) permet à l'attaquant d'injecter du script js / html pour attaquer les clients qui se connectent au site.

On dénote plusieurs types de XSS:

- Stored (Ce type permet à l'attaquant d'injecter le script de manière permanente sur le site web)
- Reflected (Moins critique que les stored, l'injection doit être faite directement sur le client victime)
- Dom based



# CROSS-SITE REQUEST FORGERY

- DESCRIPTION DE LA VULNÉRABILITÉ
- EXEMPLE CONCRET









# SERVER SIDE



# SQL INJECTION

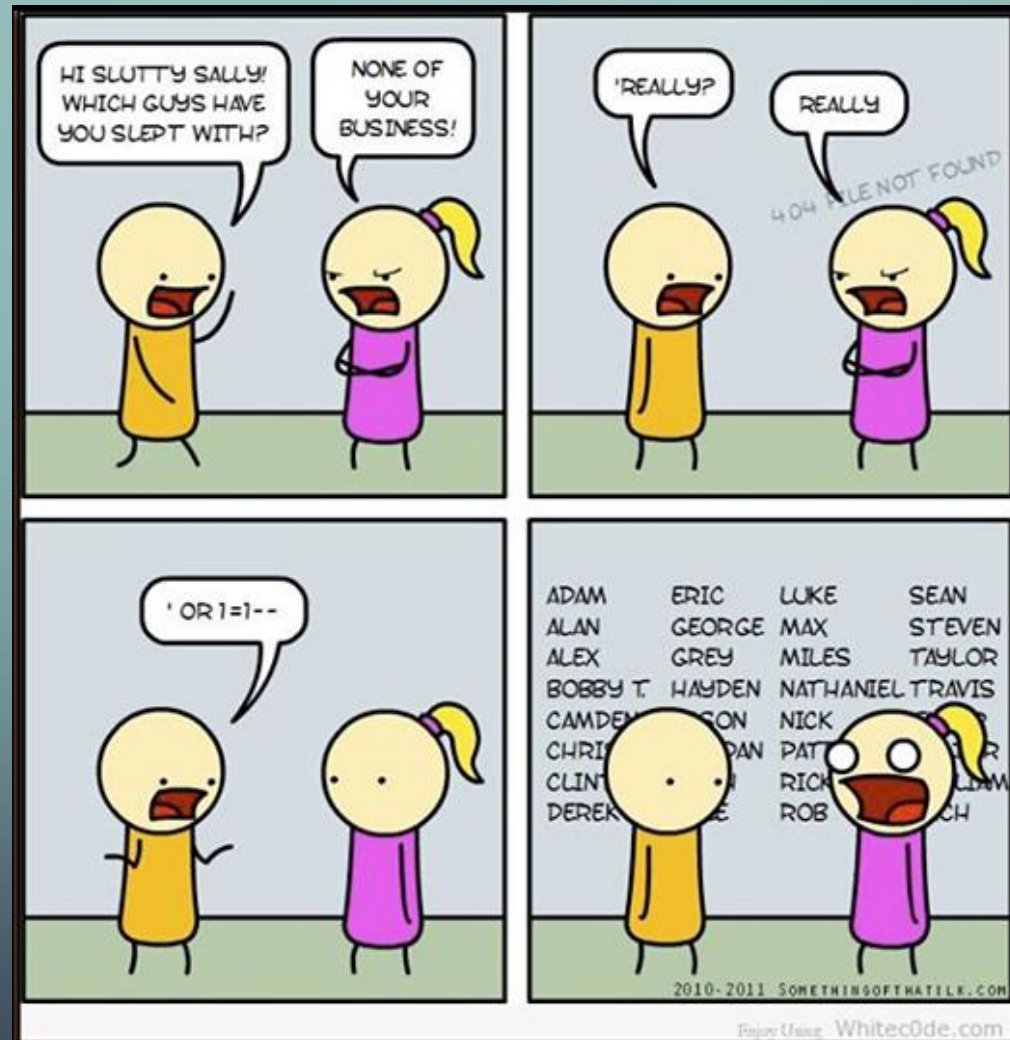
ID=99999'UNION(SELECT(1),VERSION(),3)--+

Username is : 5.5.52-cll

WTF !



# EXEMPLE CONCRET







# REMOTE CODE EXECUTION

LA RCE EST UNE VULNÉRABILITÉ HAUTEMENT CRITIQUE, ELLE PERMET D'EXECUTER DU CODE DE MANIÈRE ARBITRAIRE DIRECTEMENT SUR LE SERVEUR VICTIME !

CA PEUX ÊTRE DU PHP PAR EXEMPLE ET VOIRE MÊME DES COMMANDES SYSTÈMES.

# PRIVILEGE ESCALATION



# DESCRIPTION DE LA VULNÉRABILITÉ

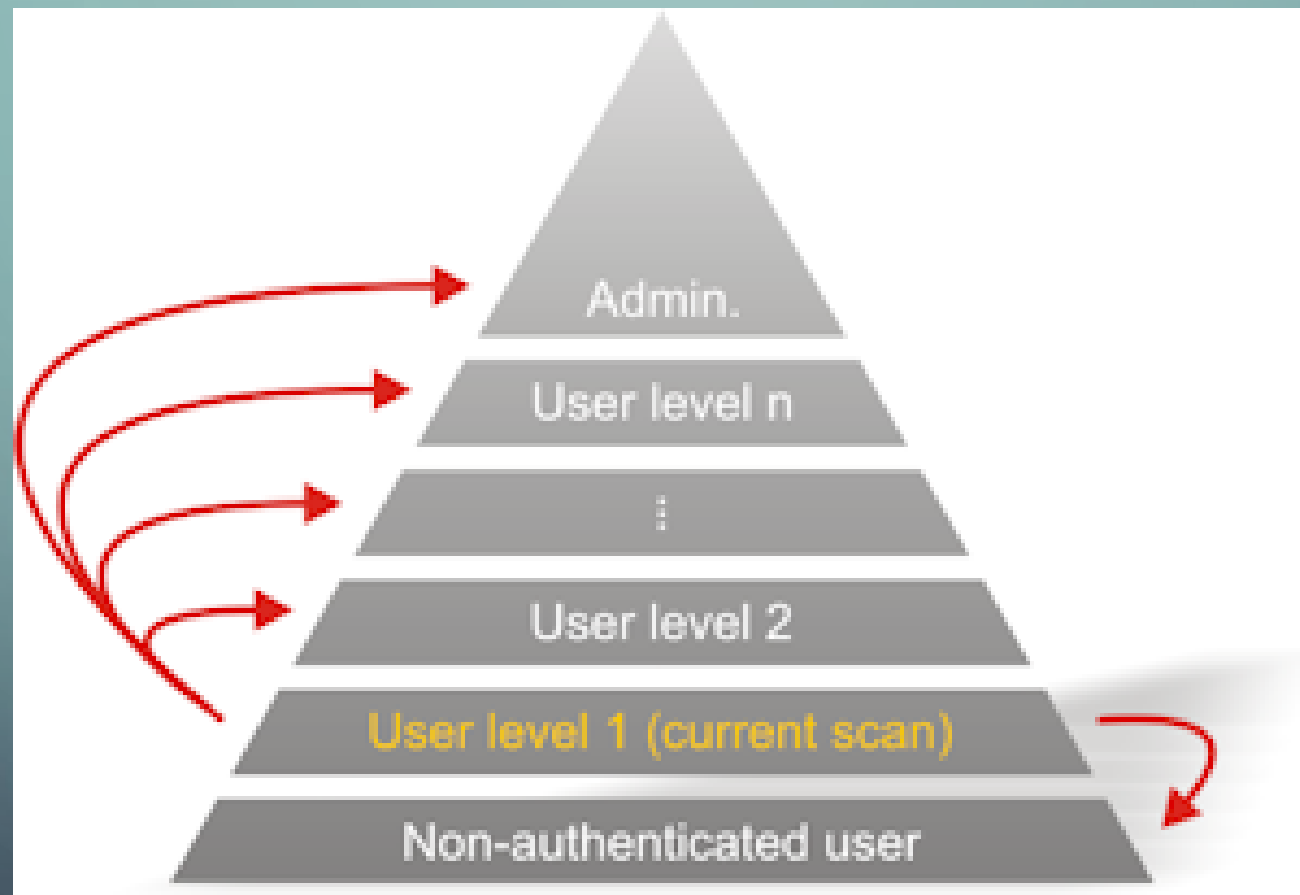
Une escalade de privilèges (Privilege Escalation) permet tout simplement de faire des choses sous un autre user sans pour autant avoir les droits.

Il y a deux types de privilege escalation:

- Vertical Privilege Escalation (en gros, on augmente ses droits en exécutant des commandes admin par exemple)
- Horizontal Privilege Escalation (plus commune, elle permet d'avoir accès à des choses du même niveau que votre compte)



# EXEMPLE CONCRET





# FILE INCLUSION

UNE FILE INCLUSION PERMET, COMME SON NOM L'INDIQUE, D'INCLURE DES FICHIERS DE MANIÈRE ARBITRAIRE !

A NOTER QU'IL EXISTE DEUX TYPES:

- REMOTE FILE INCLUSION (EN GROS, ON VA INCLURE UN FICHIER DISTANT SUR LA VICTIME)
- LOCAL FILE INCLUSION (TOUT SIMPLEMENT INCLURE UN FICHIER LOCAL !)



# FULL PATH DISCLOSURE

LA FULL PATH DISCLOSURE ...

TELLEMENT COMMUNE QUE CA EN DEVIENT VEXANT !

EN SOIT, ON NE PEUT GÉNÉRALEMENT RIEN EN FAIRE SI ELLE EST SEULE MAIS, COUPLÉE À UNE AUTRE VULNÉRABILITÉ, ELLE PEUT VITE DEVENIR CRITIQUE !

ELLE PERMET TOUT SIMPLEMENT D'AVOIR LE CHEMIN COMPLET D'UN FICHIER PRÉSENT SUR LE SITE WEB (GÉNÉRALEMENT), GRÂCE À ELLE ON POURRA FAIRE DE LA PRISE D'INFORMATIONS, DE MANIÈRE À DRESSER UN SCHÉMA DU SITE WEB OU, COUPLÉE À UNE LFI, ELLE NOUS PERMETTRA DE RÉCUPÉRER DIVERS FICHIERS DU SITE !

# SERVER MISCONFIGURATION



# DESCRIPTION DE LA VULNÉRABILITÉ









Le nom parle de lui-même ! Ca regroupe tout simplement les erreurs possibles des admin-sys/développeurs et CIE.

On peut par exemple penser à l'oubli de mettre un htaccess/index dans un dossier, nous permettant donc d'en voir tous les fichiers !

Comme pour les full path disclosure, le but est – ici aussi – de récupérer un maximum d'informations sur notre victime !

# EXEMPLE CONCRET

## Index of /pmwiki

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">Parent Directory</a>	27-Feb-2008 00:02	-	
 <a href="#">cookbook/</a>	17-Nov-2007 18:33	-	
 <a href="#">docs/</a>	17-Nov-2007 18:33	-	
 <a href="#">local/</a>	17-Nov-2007 18:33	-	
 <a href="#">pmwiki.php</a>	13-Nov-2007 22:30	77k	
 <a href="#">pub/</a>	17-Nov-2007 18:33	-	
 <a href="#">scripts/</a>	17-Nov-2007 18:33	-	
 <a href="#">wikilib.d/</a>	17-Nov-2007 18:33	-	



# PHP TYPE JUGGLING

Loose comparisons with ==												
	TRUE	FALSE	1	0	-1	"1"	"0"	"-1"	NULL	array()	"php"	""
TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE
FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE
1	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
0	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE
-1	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
"1"	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
"0"	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
"-1"	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
NULL	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE
array()	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE
"php"	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
""	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE



# DESCRIPTION DE LA VULNÉRABILITÉ

Ce n'est pas une vulnérabilité à proprement parler mais surtout une erreur de la part du développeur ...

PHP n'ayant pas besoin qu'on définisse explicitement le type de variable que l'on crée, on peut très bien se retrouver à comparer une string avec un int...

Le problème vient à partir du moment où le développeur va se contenter de comparer la valeur et non le type !



# GOOGLE DORKS

Quelle magnifique invention que les dorks google!

Tout simplement, une simple recherche google sur un site spécifique nous permettra d'avoir tout ce qui a été répertorié, cela inclut n'importe quels fichiers/dossiers !

# BUG BOUNTY PROGRAMS

Pour pallier les risques, et afin de garantir une sécurité maximale à leurs applications, les sites proposent maintenant des 'bug bounty' !

Le bug bounty, c'est tout simplement une autorisation explicite pour les hackers de chercher dans l'appli différentes vulnérabilités dans la limite de leurs autorisations !

En contrepartie, il peuvent vous récompenser avec de l'argent/des cadeaux et vous inscrire sur leur Hall Of Fame !