Sécurité des applications Web

Sommaire

1. Vous avez dit "menace" ?
2. Architecture des applications Web
3. Notions protégées
4. Intégration de la sécurité
5. Serveur HTTP
6. Base de données
7. Applications

Vous avez dit "menace" ?

Web

- devenu incontournable → présence obligatoire ;
- très utilisé → plus un domaine d'initiés (ArpaNet) ;
- partie la plus complexe à sécuriser car très exposée ;
- attaques fréquentes (Sony, ministère des finances, …)

Web

- petite structure également visée :

  www.tala-informatique.fr → 45 à 65 attaques / jours

sécurité des applications
=
élément à prendre en compte !

Aperçu des mails de l'adminstrateur de tala-informatique.fr

[Fail2Ban] SSH: banned 131.161.52.3 from s18869...	Fail2Ban	26/04/2017 01:18
[Fail2Ban] SSH: banned 125.107.253.6 from s1886...	Fail2Ban	26/04/2017 01:23
[Fail2Ban] SSH: banned 180.154.13.41 from s1886...	Fail2Ban	26/04/2017 01:30
[Fail2Ban] SSH: banned 121.22.25.152 from s1886...	Fail2Ban	26/04/2017 01:34
[Fail2Ban] SSH: banned 51.254.222.83 from s1886...	Fail2Ban	26/04/2017 02:56
[Fail2Ban] SSH: banned 36.107.110.229 from s188...	Fail2Ban	26/04/2017 03:08
[Fail2Ban] SSH: banned 187.188.209.172 from s18...	Fail2Ban	26/04/2017 04:35
[Fail2Ban] SSH: banned 39.155.136.34 from s1886...	Fail2Ban	26/04/2017 05:13
[Fail2Ban] SSH: banned 62.181.46.156 from s1886...	Fail2Ban	26/04/2017 05:31
[Fail2Ban] SSH: banned 114.4.68.177 from s18869...	Fail2Ban	26/04/2017 05:52
[Fail2Ban] SSH: banned 186.178.172.151 from s18...	Fail2Ban	26/04/2017 06:10
[Fail2Ban] SSH: banned 123.164.164.211 from s18...	Fail2Ban	26/04/2017 08:13
[Fail2Ban] SSH: banned 61.166.31.22 from s18869...	Fail2Ban	26/04/2017 08:33
[Fail2Ban] SSH: banned 222.67.44.46 from s18869...	Fail2Ban	26/04/2017 08:53
[Fail2Ban] SSH: banned 175.139.127.58 from s188...	Fail2Ban	26/04/2017 09:01
[Fail2Ban] SSH: banned 183.93.223.209 from s188...	Fail2Ban	26/04/2017 09:43
[Fail2Ban] SSH: banned 47.88.6.245 from s188692...	Fail2Ban	26/04/2017 09:48
[Fail2Ban] SSH: banned 218.78.213.153 from s188...	Fail2Ban	26/04/2017 10:22
[Fail2Ban] SSH: banned 41.82.38.65 from s188692...	Fail2Ban	26/04/2017 10:25

Entamons notre tour du monde par les pays de l'est ! ...

• 178.158.236.52 :
person: Vadim A. Alimov
address: 23, Krupskoy str., Boyarka, Ukraine
phone: +38 067 8270577

• 84.54.248.122 :
person: Alexander Ivanov
address: 290, Myra st., 355000, Stavropol, Russia
phone: +7 8652 249595

• 178.46.15.255 :
role: Uralsvyazinform Perm Administration Staff
address: 11, Moskovskaya str., Yekaterinburg, 620014
address: Russian Federation

… continuons avec la Chine ! ...

• 116.252.34.161 :
person: Bin Deng
address: Guangxi data comm.Bureau
address: 35 Minzhu Road, Nanning city, China

• 125.113.3.24 :
role: CHINANET-ZJ Jinhua
address: No.155 Xishi street,Jinhua,Zhejiang.321000 China

• 49.140.100.96 :
descr: Jilin University
descr: Changchun, Jilin 130012, China

… poursuivons avec l'Amérique du Sud ! ...

• 152.204.9.16 :
owner: COLOMBIA TELECOMUNICACIONES S.A. ESP
person: Grupo de Administradores Internet
address: Transversal, 60, 114 A, 55
address: 571111 - BOGOTA DC – CU

• 181.211.180.149 :
owner: CORPORACION NACIONAL DE TELECOMUNICACIONES - CNT EP
ownerid: EC-ANSA-LACNIC
responsible: Evelin Gavilanes
address: Jorge Drom y Gaspar de Villaroel, 954, 1 er Piso
address: 3110 - Quito – ECUADOR

… poursuivons avec l'Amérique du Sud ! ...

• 201.177.40.188 :
owner: Telefonica de Argentina
ownerid: AR-TEAR7-LACNIC
responsible: José Luis Pérez Elias
address: AV. ING. HUERGO, 723, GERENCIA DE REQUERIMIENTOS JUDICIALES
address: 1065 - Buenos Aires – CF

• 201.217.142.186 :
nic-hdl: ANU
person: ANTELDATA ANTEL URUGUAY
e-mail: ipadmin@ANTEL.NET.UY
address: Mercedes, 876, P. 2
address: 11100 – Montevideo, URUGUAY

… au tour de l'Inde ! ...

• 117.222.98.129 :
netname: BB-Multiplay-General
descr: Broadband Multiplay Project, O/o DGM BB, NOC BSNL Bangalore
country: IN

• 37.49.224.115 :
org-name: Estro Web Services Private Limited
org-type: OTHER
address: H. No. 1, Mangu Panna
address: Tatesar wala Rasta, Village - Jaunti
address: Delhi - 110081, India

… au tour de l'Inde ! ...

• 122.174.183.87 :
person: Network Administrator for ABTS TN
address: ABTS Tamilnadu
address: 101,Santhome High Road,Chennai,Tamilnadu

• 110.227.244.22 :
netname: GPRS-Subscribers-in-East
descr: BCL EAST
descr: 7th Floor,Infinity Towers,
descr: salt Lake,Sector-V,Electronic Complex
descr: Kolkata

… à l'origine les pirates fréquentaient beaucoup les îles ! ...

• 80.82.65.21 :
org-name: Quasi Networks LTD.
address: Suite 1, Second Floor
address: Sound & Vision House, Francis Rachel Street
address: Victoria, Mahe, SEYCHELLES

• 185.10.68.229 :
org-name: Flokinet Ltd
org-type: LIR
address: Suite Number 2, Olivier Maradan Building, Olivier Maradan St
address: Victoria
address: SEYCHELLES

… peut-être convoitaient-ils une place dans le Valhalla ? ...

• 84.216.90.249 :
person: Fredrik Thorslund
address: Spray Network AB
address: Ostermalmsgatan 87
address: SE-10244 Stockholm

• 81.8.184.211 :
person: Gosta Vistemar
address: Industrivagen 7
address: 912 34 VILHELMINA
address: SWEDEN

… peut-être convaitaient-ils une place dans le Valhalla ? ...

• 84.202.187.239 :
netname: NO-EAB-CABLE
descr: Bergen Loddefjord, Norway

• 62.16.175.14 :
netname: NO-EAB-CABLE-BARUM
address: Canal Digital Kabel TV AS
descr: Bærum Kolsås, Norway

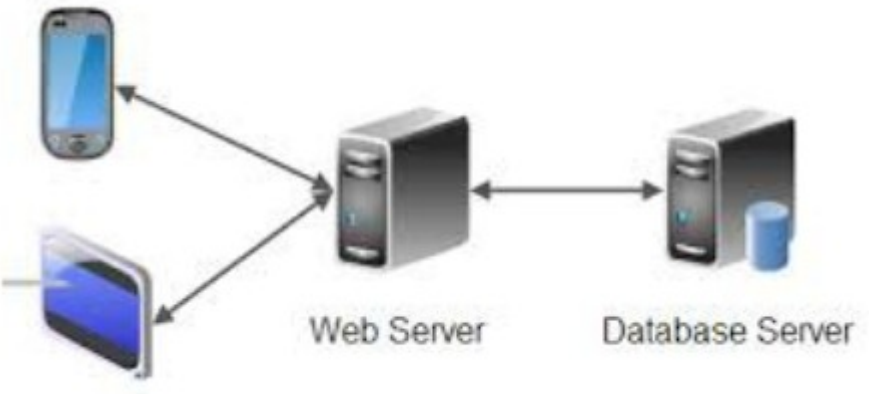
• 137.163.145.226 :
netname: HELNET
person: Simo Volanen
address: City of Helsinki, Finland

Architecture des applications Web

N-tiers

- Les applications Web sont basées sur une architecture N-tiers ;

- Séparation entre :
  - → la présentation des données ;
  - → le traitement métier des données ;
  - → l'accès aux données persistantes.



N-tiers

- Couche présentation :
  → obligatoirement en HTML / CSS / Javascript ;

- Couche métier :
  → PHP / ASP ;
  → Java ;

- Couche persistance :
  → SQL ;
  → No SQL;

Notions protégées

L' identification est l'information qui permet d'indiquer qui une personne prétend être :

- nom d'utilisateur ;
- empreintes digitales ;
- analyse faciale ;
- analyse rétinienne ;
- ...

L'authentification est l'information qui vient valider l'identification :

- faible → mot de passe ;
- forte → combinent une chose possédée et une chose connue (eg. carte bancaire / code personnel).

L'autorisation est l'information permettant de déterminer à quelles ressources de l'entreprise une personne authentifiée est autorisée à accéder et les actions qu'elle peut entreprendre.

La confidentialité est l'ensemble des mécanismes permettant qu'une communication de données reste privée entre un émetteur et un destinataire :

La cryptographie ou le chiffrement, IPsec (couche 3 OSI) et SSL (couche 7 OSI) sont les seules solutions fiables pour garantir la confidentialité des données.

L'intégrité est l'ensemble des mécanismes qui permettent d'assurer qu'une information n'a pas été modifiée.

La disponibilité est le fait de garantir que les ressources de l'entreprise sont accessibles :

- réseau ;
- bande passante ;
- ...

La non-répudiation permet de garantir qu'un message a bien été échangé entre un émetteur et un destinataire.

La traçabilité est l'ensemble des mécanismes permettant de retrouver les opérations réalisées sur les ressources de l'entreprise constituant le SI.

Intégration de la sécurité

Assurer la sécurité

- Pas un métier inné :
  - ➔ années d'expérience ;
  - ➔ documentation ;
  - ➔ écriture de code / test.

- Leitmotiv :
  - ➔ intégration dès le début du projet ;
  - ➔ attention constante (discipline de développement, respect des règles) ;
  - ➔ veille très importante ;
  - ➔ tests sur panel large.

Assurer la sécurité

- Faire attention aux prestataires externes (se renseigner)

- Si application compromise :
  - ➔ audit ;
  - ➔ ré-installation des serveurs ;
  - ➔ coût financier important ;
  - ➔ impact sur l'image de marque.

Quelles sont les menaces ?

Failles souvent exploitées

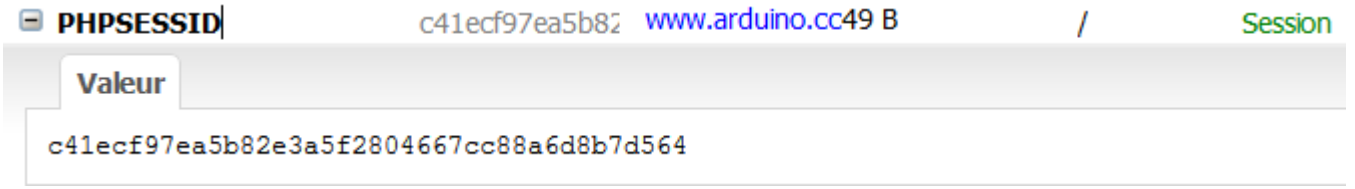
- Gestion de l'authentification et des droits ;
- Cross-Site :
  Scripting (XSS) / Tracing (XST) / Request Forgery (CSRF) ;
- Injection SQL ;
- Fichiers interprétables ;
- Injection de commandes systèmes ;
- Exposition d’informations via les messages d’erreur ;
- Race condition.

Gestion de l'authentification et des droits

Cas de figure :

Vol de session grâce aux cookies ou élévation de droits

PHPSESSIONID



JSESSIONID

Cross-Site Scripting (XSS)

Objectif : faille permettant d'injecter du code dans une page pour provoquer une action sur les navigateurs cibles

• XSS réfléchi (ou non permanent) :
utilisé pour faire du hameçonage (phishing) et requiert de l'ingénierie sociale pour fonctionner !

• XSS stocké (ou permanent) :
lorsque le code est injecté en base de données puis représenter sans traitement.
Attaque puissante qui ne requiert pas l'ingénieurie sociale pour fonctionner !

Cross-Site Scripting (XSS)

Contre-mesure :

- Traitement quasi systématique du code HTML produit par l'application avant envoi au navigateur ;

- PHP
  → htmlspecialchars(), htmlentities(), strip_tags()
  → plus simple = *HTML Purifier*

Cross-Site Tracing (XST)

Objectif : mettre à profit la méthode TRACE du protocole HTTP pour contourner les CORS (Cross-origin resource sharing).

Cas de figure : idem XSS

Contre-mesure : idem XSS

Cross-Site Scripting Request Forgery (CSRF)

Objectif : permettre à un utilisateur lambda de faire une action qui nécessite normalement une élévation de droit.

Cas de figure :

- Site utilisant un système d'authentification vulnérable ;
- Requête HTTP falsifiée envoyée par mail à un utilisateur possédant les droits ;
- L'utilisateur ouvre le mail et exécute la requête (souvent à son insu → balise "src").

Cross-Site Scripting Request Forgery (CSRF)

Contre-mesure :

- confirmation utilisateur pour les actions critiques
  → alourdissement des procédures ;

- Utilisation de jetons de validité dans les formulaires
  → vérification OTT (One Time Token) côté serveur ;

- Éviter HTTP GET pour effectuer des actions
  → forgery JavaScript toujours possible !

- Utiliser des référents
  → bloque la requête si la valeur de référence est différente de la page d'où il doit théoriquement provenir.

Injection SQL

Objectif : injecter un morceau de requête SQL dans la requête en cours

Cas de figure :

- requête utilisant des valeurs utilisateurs comme paramètres d'entrées
- la requête originale est détournée pour extraire de l'information supplémentaire ou, pire, supprimer des données

Injection SQL

Contre-mesure : Traitement des entrées utilisateur avant envoi au SGBD ;

- PHP
  → addslashes(), mysqli_real_escape_string() ,
  → utilisation de PDO

Injection de commandes systèmes

Objectif : détournement de la fonction shell_exec() ou exec() pour exécuter des commandes non désirées !

Cas de figure : les entrées utilisateur sont utilisées pour effectuer des traitements sur le système.

Contre-mesure : Traitement des entrées utilisateur avant utilisation

Fichiers interprétables

Objectif : détournement de la fonction include ou require pour inclure du contenu exécutable

Cas de figure : les entrées utilisateur sont utilisées pour inclure des fichiers.

Contre-mesure : Traitement des entrées utilisateur avant utilisation.

Exposition d'informations via les messages d'erreur

Objectif : utilisation des sorties d'erreur pour découvrir des informations sur l'architecture de l'application.

Cas de figure : mauvaise configuration de l'interpréteur, du serveur HTTP ou du SGBD qui laisse transparaître des informations sur les versions, modules ou fichiers utilisés dans l'application Web.

Situation de compétition (race condition)

Objectif : utiliser le laps de temps qui s'écoule entre la réussite de l'authentification et l'action nécessitant cette authentification.

Cas de figure : l'attaquant profite de la "lenteur" du système ou de la connexion de la victime pour effectuer l'action avant elle.

Contre-mesure :

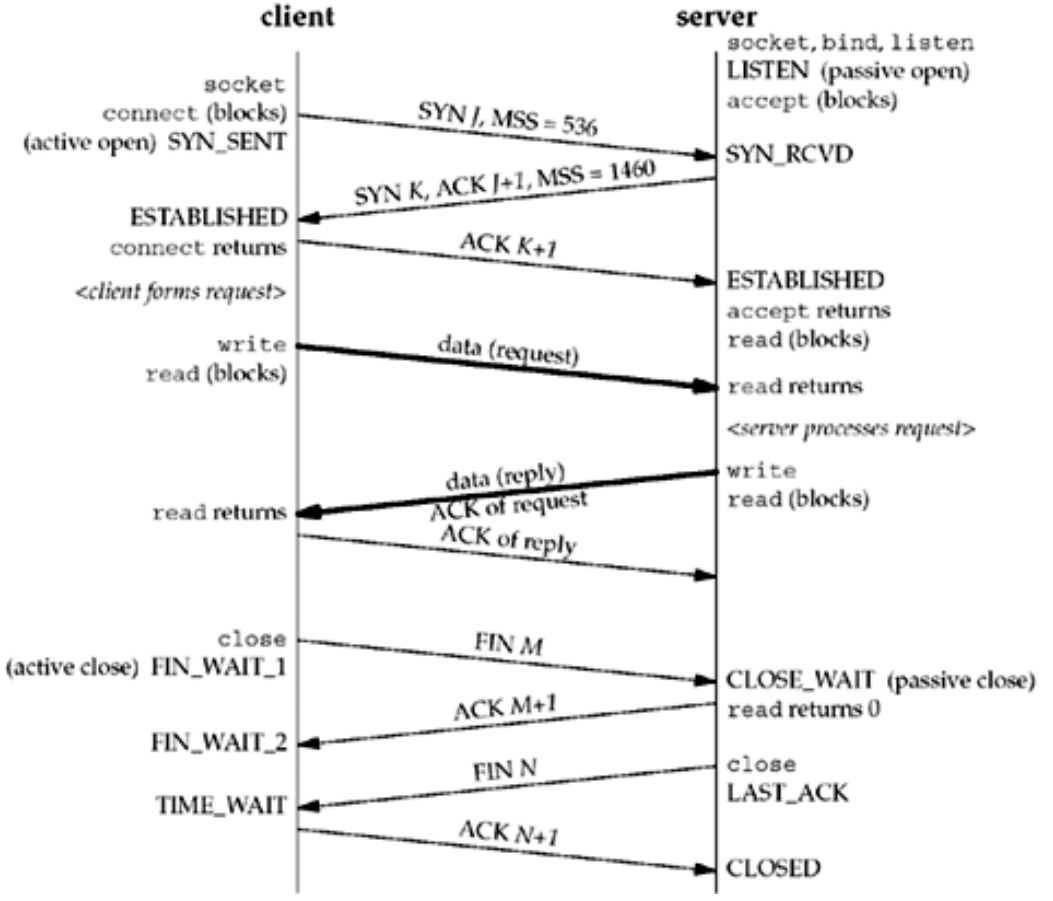
→ ne pas séparer l'authentification de l'action,

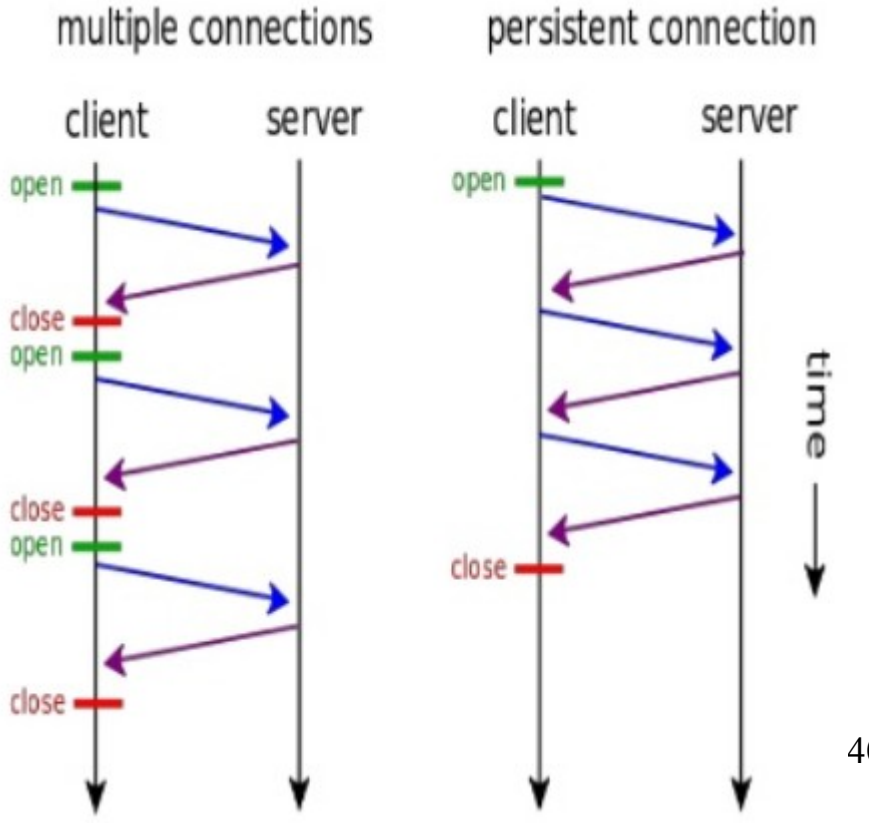
→ utiliser des attributs permettant une identification unique (IP + port)

Serveur HTTP

Performances importantes pour l'UX :

- Réutilisation au maximum des connexions TCP ;
  → réduit la consommation CPU
  → accélère l'échange





Performances importantes pour l'UX :

- Utilisation des capacités POSIX du système :
  → environnement / modules "Thread Safe"
  → diminue la consommation mémoire

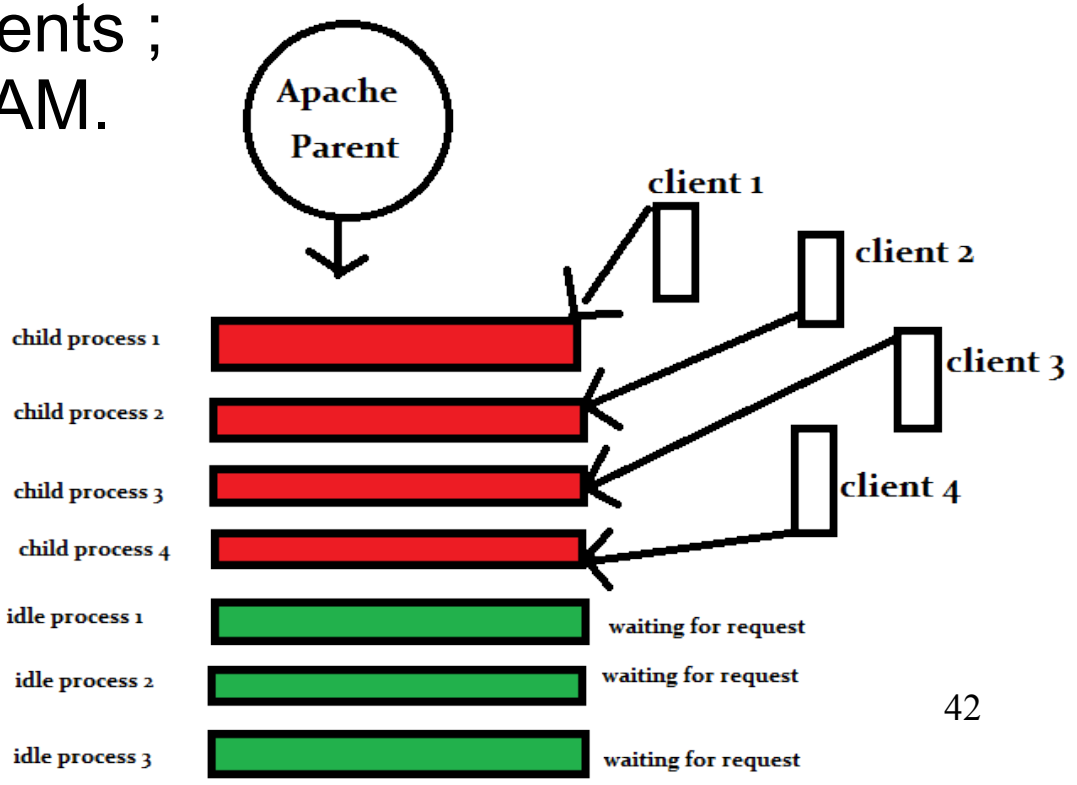
Par exemple, HTTPD possède 3 modes de fonctionnement POSIX :

→ mpm_prefork (processus lourd = fork C)
→ mpm_worker (processus léger = thread)
→ mpm_event (mélange des deux précédents)

ATTENTION
→ **compromis vitesse / consommation / sécurité !**

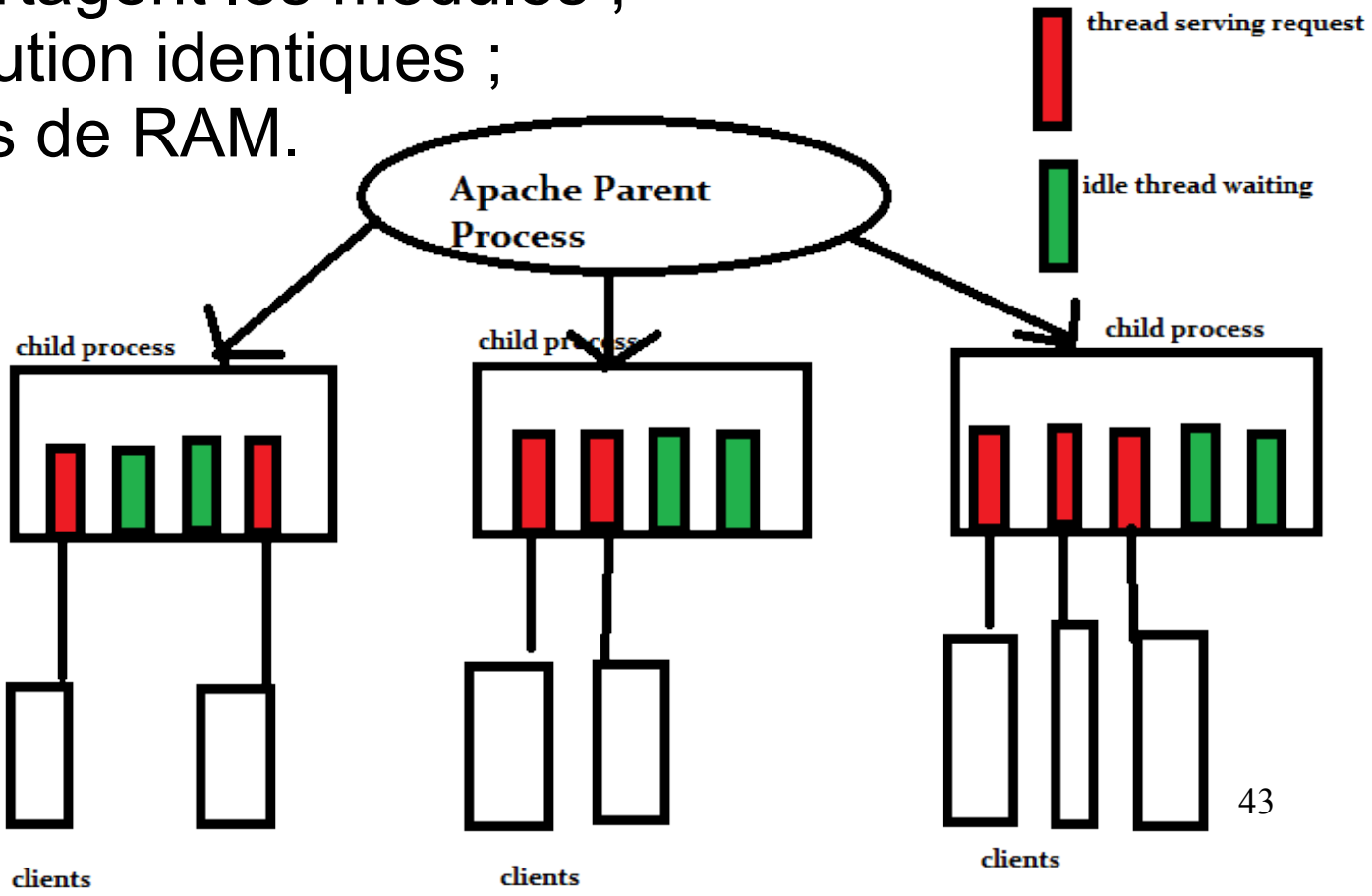
mpm_prefork

- thread safe ;
- un processus lourd par client ;
- chaque processus embarque ces modules (interpréteurs, etc…) ;
- contextes d'exécution différents ;
- consomme beaucoup de RAM.



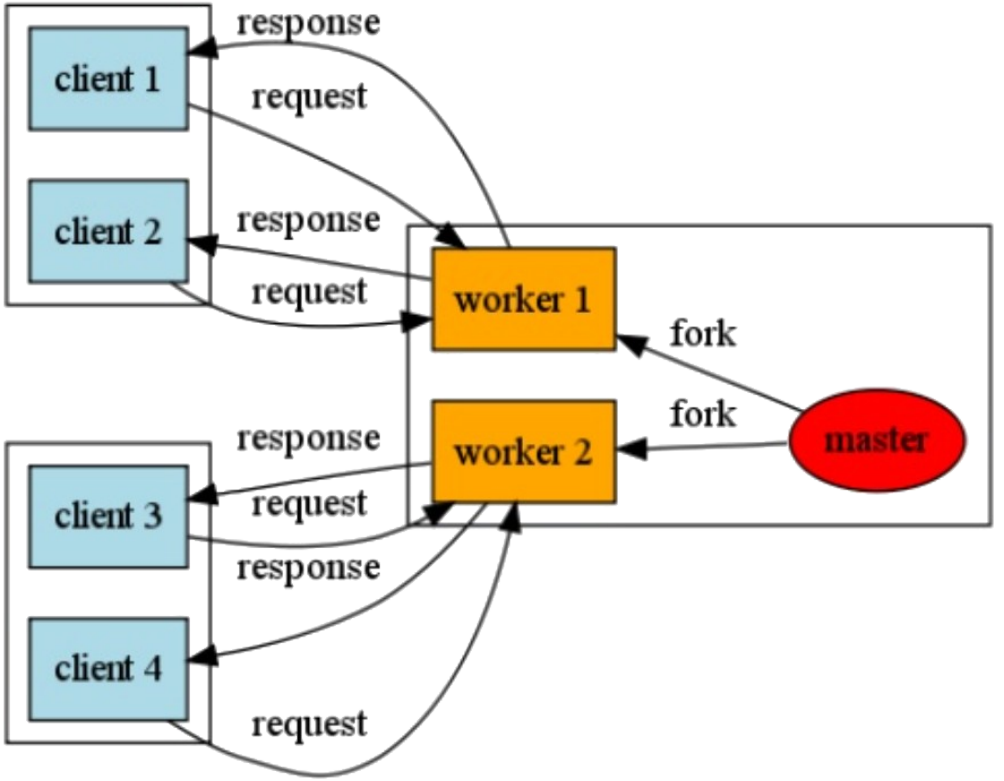
mpm_worker

- pas thread safe (attention aux modules utilisés) ;
- un processus léger par client ;
- les processus partagent les modules ;
- contextes d'exécution identiques ;
- consomme moins de RAM.



mpm_event

- quasi idem mpm_worker ;
- différents clients partagent les même threads ;
- utile en sessions TCP persistentes longues ;
- impossible en SSL → sticky connection.



Ne pas donner trop d'informations :

- Eviter les signatures dans les entêtes :
  httpd.conf
  → ServerToken Prod
  → ServerSignature Off
  php.ini
  → expose_php = Off
  → error_reporting = E_ALL & ~E_DEPRECATED & ~E_STRICT

Réponse voir le code source

Cache-Control	no-store, no-cache, must-revalidate
Connection	close
Content-Length	1654
Content-Type	text/html; charset=UTF-8
Date	Fri, 28 Apr 2017 15:51:55 GMT
Expires	Thu, 19 Nov 1981 08:52:00 GMT
Pragma	no-cache
Server	Apache/2.2.15 (CentOS)
X-Powered-By	PHP/7.0.18

Réponse voir le code source

Cache-Control	no-store, no-cache, must-revalidate
Connection	close
Content-Length	1654
Content-Type	text/html; charset=UTF-8
Date	Fri, 28 Apr 2017 15:55:33 GMT
Expires	Thu, 19 Nov 1981 08:52:00 GMT
Pragma	no-cache
Server	Apache

Base de données

- changer de mot de passe par défaut :

*grant all privileges on *.* to root@127.0.0.1 identified by "password";*

- Ne pas autoriser les connexions distantes :

 # iptables -I INPUT 2 -i $eth -p tcp --dport 3306 -j DROP

Exposer la base de données directement sur Internet serait trop dangeureux et permettrait les attaques de type brute force !

- Ne pas autoriser les connexions distantes :

Si un prestataire externe requiert un accès direct à la base de données (mise à jour, bux fix, …) → utiliser un VPN !

Si impossible, utiliser *fail2ban* mais attention, ne protège pas les interface d'administration web (eg. *phpMyAdmin*) :

```
[mysql]
enabled  = true
port     = 3306
filter   = mysql
logpath  = /var/log/mysql/error.log
maxretry = 3
```

- Ne pas autoriser les connexions distantes :

Seule défense pour les interfaces d'administration web (eg. *phpMyAdmin*) → DPI (Deep Packet Inspection) :

→ Out of the box : SNORT + Barnyard2 + snort_inline (IPS)

→ «Maison» : *SOCKET_RAW* en *PHP*
pack() / unpack() + *hex2bin*

*Cip_ver_len/Ctos/ntot_len/nidentification/nindic_frag_offset/Cttl/Cprotocol/nchecksum/Nsrc_addr/Ndst_addr/H*payloadCip_ver_len/Ctos/ntot_len/nidentification/nindic_frag_offset/Cttl/Cprotocol/nchecksum/Nsrc_addr/Ndst_addr/H*payload*

- Précepte quasi-biblique :

« tout ce qui ne sert pas doit être supprimé ! »

→ supprimer les comptes inutiles :

Pour les plus paranoïaques, on peut renommer le compte administrateur.

→ supprimer les bases exemple :

Attention aux bases systèmes...

- Activer les logs et les externaliser (analyse post-mortem) :

Dans «/etc/mysql/my.cnf» :

general_log_file = /var/log/mysql/mysql.log
general_log = 1

Puis (* si logrotate) :

 # rsync -avz -e ssh /var/log/mysql/mysql.log $srv:$path*

Pour se prémunir de :

PURGE BINARY LOGS TO '/dev/null' BEFORE NOW();
PURGE MASTER LOGS TO '/dev/null' BEFORE NOW();

(Idem → history -c)

- Eviter les attaques par débordement de tampon :

→ exécuter le processus avec un compte de service

mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/bash

→ utiliser SELinux : nécessite une forte connaissance du service à protéger

 # semanage port -a -t mysqld_port_t -p tcp 3307
 # semanage fcontext -a -t mysqld_db_t "/datadir(/.)?"*
 # restorecon -Rv /datadir
restorecon reset /datadir context unconfined_u:object_r:default_t:s0→
unconfined_u:object_r:mysqld_db_t:s0
….

→ *utiliser des containers (pertes de performances)*

- Eviter les attaques par débordement de tampon :

→ *utiliser des conteners (pertes de performances sur Docker)*

MACHINE	RW (TRANS/SEC)
Stock instance running on host	12686
Docker Volume –net=host	12201
Docker Hostdir –net=host	12180
Docker Datacontainer –net=host	12233
Docker Volume –net=bridged	11698
Docker Hostdir –net=bridged	11644
Docker Datacontainer –net=bridged	11662

→ *chroot pénible :*

chroot=/services/chrooted-mysql puis jouer avec «ldd»;

- Gestion des données :

→ chiffrer les données stockées (SHA256 ou R/DSA) ;

→ éviter les «*ON CASCADE*» ;

→ *utiliser des procédures stockées (user input).*

- Faire les mises à jours «critiques» :

→ attention à la matrice de compatibilité

Applications

- Intégrer la sécurité à la conception :

→ trop de chefs de projet «oublient» l'aspect sécuritaire ;
→ difficile de repasser sur une application déjà développée ;
→ confiance utilisateur / image de marque.

- Suivre les recommandations basiques :

→ accès base PDO ou driver propriétaire
→ respect des design patterns (DAO, MVC, etc…)

- Utiliser des rôles dans l'application (éclater les pouvoirs)

- Utiliser un BRMS (Business Rules Management System) comme Drools (*plugin Eclipse*)

- Traquer les actions, le fameux «KIFEKOI»

- Traquer les connexions en interne :

→ ne pas faire «confiance» aux mécanismes internes ;
→ gestion des ID de session / IP+port ;

- Faire une implémentation «full-stack» du protocole HTTP :

→ permet aux outils des externes une analyse comportementale;
→ proxyfication plus simple.
→ gestion des session multiples / SSE (push server)