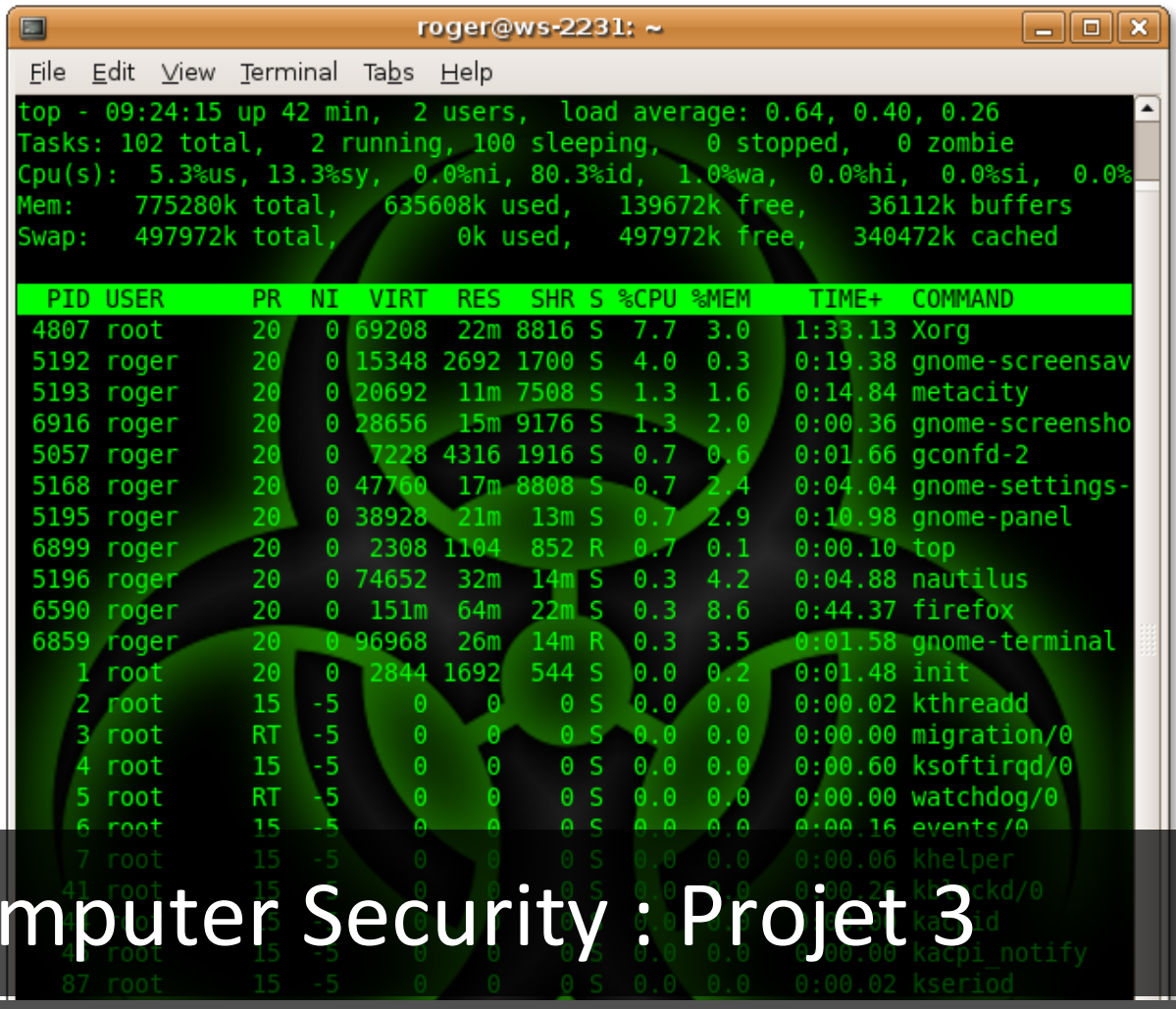

Computer Security : Projet 3

Quang Vinh Pham & Alexandre Balon-Perin

Groupe 15

Ecole Polytechnique de l'ULB
1er Master en Technologie de l'information et de la communication

Table of Contents

Introduction

Le but de ce projet est d'obtenir un site web PHP/MySQL dont le serveur est Apache résistant à un maximum de types d'attaques.
On peut distinguer les écoutes des messages transitant entre le serveur et le client, les injections SQL ainsi que les violations de la base de données ou de la machine du client par des routines écrites en javascript.
Pour se défendre de ces attaques, on peut compter respectivement sur l'OpenSSL, le blindage des champs de formulaire et le cryptage des entrées dans la base de données.

Moyens de protection

OpenSSL

L'OpenSSL va permettre de créer un canal de communication sécurisé entre le client qui utilise le site web et le serveur qui héberge celui-ci. Via ce canal, l'utilisateur pourra envoyer des données vers le serveur en toute sécurité. En effet, OpenSSL repose sur le principe du chiffrement asymétrique, c'est-à-dire que le serveur dispose d'une clé privée qu'il est le seul à connaître et que tous les utilisateurs ont accès à la clé publique correspondante. La clé publique se trouve en fait dans le certificat que les utilisateurs acceptent (manuellement ou automatiquement selon que le certificat soit reconnu par des organismes ou auto-singé) en se connectant à la version sécurisée HTTPS du site. Le protocole OpenSSL cryptera toutes les données envoyées par les utilisateurs avec cette clé publique, données qui ne pourront être lues que par le serveur.

Blindage des champs

Le blindage des champs nous permet de nous protéger des injections SQL. En effet, au lieu de rentrer de simples login ou password, des gens malintentionnés peuvent entrer des requêtes SQL qui seront interprétées par la base de données et qui peuvent avoir des conséquences néfastes. On peut penser à la connexion au site sans s'être enregistré au préalable, l'usurpation d'identité, trouver le mot de passe d'un autre utilisateur, modifier la base de données,…
Pour se protéger, il faut blinder les champs, c'est-à-dire ne permettre aucune syntaxe qui s'apparenterait à une requête. Pour ce faire, on va utiliser des méthodes simples mais efficaces, comme mettre un \ devant chaque ', ce qui aura pour effet que la requête sera compris comme un simple String et n'aura aucun effet sur la base de données.
Le blindage intervient également pour se protéger des scripts malintentionnés Javascript. En convertissant tout symbole en un caractère spécial HTML, on évite

la mauvaise surprise d'une injection de code javascript. Le blindage que nous avons utilisé est présenté dans la fonction de protection du code.
Un moyen simple d'éviter l'injection est de limiter le nombre de caractères autorisés.

Cryptage des données

Si malgré tous ces efforts, un adversaire arrivait tout de même à accéder à la base de données, il faudrait que celle-ci soit cryptée afin qu'il ne puisse accéder à aucune information sensible. Ce chiffrement se ferait par un chiffrement symétrique de type DES dont on garderait la clé quelque part bien sécurisé.
Un autre réflexe qui a son importance est de hasher automatiquement les mots de passe. En effet, ceux-ci ne devant pas être récupérés dans leur forme originale, on peut se permettre de comparer les hashs pour authentifier l'utilisateur.

Implantation du protocole OpenSSL

Afin de configurer le serveur Apache à l'OpenSSL et au HTTPS, il faut suivre quelques étapes. Tout ce qui suit se passe en ligne de commande.

- On passe en mode administrateur (sudo su)
- On entre le mot de passe approprié
- On arrête le serveur Apache (/etc/init.d/apache2 stop)
- On active le module SSL (a2enmod ssl)
- On crée un hôte virtuel pour le site en HTTPS
  (touch mondomaine.https dans /etc/apache2/sites-available
  dans mondomaine.https : on entre les lignes suivantes :
  NameVirtualHost *:443
  <VirtualHost *:443>
  …
  …
  SSLEngine On
  SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
  SSLCertificateFile /etc/ssl/certs/mondomaine.crt
  SSLCertificateKeyFile /etc/ssl/private/mondomaine.key
  </VirtualHost>)
- On ajoute l'hôte virtuel (a2ensite mondomaine.https)
- On ouvre le port 443 (on va dans le fichier /etc/apache2/ports.conf et on ajoute Listen 443)
- On crée une clé privée dans un dossier de son choix à l'aide d'une passphrase au choix (openssl genrsa –des3 –out mondomaine.key 1024)(si on veut pas de passphrase on enlève –des3)
- On fait la demande de certificat (openssl req –new –key mondomaine.key –out mondomaine.csr)
- On doit remplir quelques champs informatifs
- On crée un certificat auto-signé (openssl x509 –req –days 365 -in mondomaine.csr -signkey mondomaine.key -out mondomaine.crt)

- On installe les certificat et clé privée
  (cp mondomaine.crt /etc/ssl/certs
  cp mondomaine.key /etc/ssl/private)
- On redémarre le serveur (/etc/init.d/apache2 start)

Résultats et modifications

/etc/apache2/sites-available/mondomaine.https

```
<IfModule mod_ssl.c>
NameVirtualHost *:443
<VirtualHost *:443>
        …

        #   SSL Engine Switch:
        #   Enable/Disable SSL for this virtual host.
        SSLEngine on
        SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
        #   A self-signed (snakeoil) certificate can be created by installing
        #   the ssl-cert package. See
        #   /usr/share/doc/apache2.2-common/README.Debian.gz for more info.
        #   If both key and certificate are stored in the same file, only the
        #   SSLCertificateFile directive is needed.
        SSLCertificateFile    /etc/ssl/certs/mondomaine.crt
        SSLCertificateKeyFile /etc/ssl/private/mondomaine.key

        …

</VirtualHost>
</IfModule>
```

/etc/apache2/ports.conf

```
NameVirtualHost *:80
Listen 80

<IfModule mod_ssl.c>
    # If you add NameVirtualHost *:443 here, you will also have to change
    # the VirtualHost statement in /etc/apache2/sites-available/default-ssl
    # to <VirtualHost *:443>
    # Server Name Indication for SSL named virtual hosts is currently not
    # supported by MSIE on Windows XP.
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
```

</IfModule>

/etc/ssl/private/mondomaine.key

-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,FDBFFA44A8049626

cPB12MpU5kyb+v1Km1wgHtXiU3RtFZc3/t+7m0keA71BDGhRD5kQ4Q88q8SgrI
Eq
3JoRXuB5crBELh1VtWmBweRBkao1WTjTtHCwO5FiW8OKpec1TVcTchXqLpHq
GAnf
jJymCKmFOHaQYcHJD8rBClAtiY31K/Du8Bpe+dV3szS9TEhecK7hYAw4rpuEx/w
A
9xTC/lpj3gssIPqg0QCb3dLzAi5/47pZD/JizmhpAbZWY1XvelUuyBIA63Xu6npl
AgfuPYsEBbqZvasT+Wl8MGxtTpOmUQxnfioGSFydMUtts6CachEMeGI5YJQhsvpp
B0feWE1XAZJ5gz4/E7PdfJXdZVeCGeDiiPqiYEJfW5KvPzyCJz4WywJm3+b1ab8s
c8U4xfv4yd7Kt1yzlnD/tybDTHCktJbOOcgaytJKhqggSIW9eq+xpsJHYv5qVTko
lANQhDqss1uDqjPQDQ0b7K+5eIfsmIoaSVACIghkxavzrUPoqM/2RnNkrApz9DR8
Bp7bltg0nsHsV60I8YlGxUuggcaqfVVQ9Qsex2pSyFTihSaCkGIO/YG9JFZ/cuIA
qjz1Zxp9Y5JV8mHWwkSdn7PKCtdeUaC2+77reRzgn8o56441PD1I9EvSrNpS9/y
P
KUgDPNTEdzKX0srknxtqXS9R+Oy5hroM36T8fbATbOXpiuH7AIuLRUIPZJYdBE3
m
mVwsJvTGJUSl6A4pHvGorUVlhjq1ADuptPGOXJaTqAgwNsD5F8EuIC+rZ9fIwaaE
WhE/ZI46M9Q7R9MOkt17/WsziM43gKSNCfQxboiyNBkQ0hOJUIrcpg==
-----END RSA PRIVATE KEY-----

mondomaine.csr

-----BEGIN CERTIFICATE REQUEST-----
MIIByTCCATICAQAwczELMAkGA1UEBhMCQkUxETAPBgNVBAgTCEJydXNzZWxz
MREw
DwYDVQQHEwhCcnVzc2VsczESMBAGA1UEChMJYWx2aW4gZGV2MRYwFAYDV
QQLEw1z
ZWN1cml0eSB1bml0MRIwEAYDVQQDEwlsb2NhbGhvc3QwgZ8wDQYJKoZIhvcN
AQEB
BQADgY0AMIGJAoGBAMM6dUE7kz/Tv/hI4VM6zvGkKWEVA/X4orMK6pReL8U
K9IOw
NR4ekBdTjuLOi/JTGvNdKbRH9+eazRHMdM9/QOOR4aGLJ6ILcr/n0N4elFc9Fdk
U
4c3d9QDrKDieZNjBr2InjflKuoL4onyPoClMB9d974/Ts0hEq6WPTXXcAFpRAgM
B
AAGgFjAUBgkqhkiG9w0BCQcxBxMFYWx2aW4wDQYJKoZIhvcNAQEFBQADgYEA
D7S1
9IKDFgWDLJfioWAyHIWvaDAcJKVer9b+6Q1FFN+ML/Z5IQ+ORchEjE5nMTXDiu
ES

sOHOe/Edht4cAgaYqeCF+Sb/yRkYhPt6/cJvTmzKOqxHQswp1b+lgARIyyODpoS
H
3OCn0ZqDFECUpo5DpbBw55s+1aftmKuqn2IykIw=
-----END CERTIFICATE REQUEST-----

/etc/ssl/certs/mondomaine.crt

-----BEGIN CERTIFICATE-----
MIICXTCCAcYCCQD6QYOeHrr2vDANBgkqhkiG9w0BAQUFADBzMQswCQYDVQQ
GEwJC
RTERMA8GA1UECBMIQnJ1c3NlbHMxETAPBgNVBAcTCEJydXNzZWxzMRIwEAY
DVQQK
EwlhbHZpbiBkZXYxFjAUBgNVBAsTDXNlY3VyaXR5IHVuaXQxEjAQBgNVBAMTC
Wxv
Y2FsaG9zdDAeFw0xMDEyMTMwNDA4MDlaFw0xMTEyMTMwNDA4MDlaMHMx
CzAJBgNV
BAYTAkJFMREwDwYDVQQIEwhCcnVzc2VsczERMA8GA1UEBxMIQnJ1c3NlbHMx
EjAQ
BgNVBAoTCWFsdmluIGRldjEWMBQGA1UECxMNc2VjdXJpdHkgdW5pdDESMBA
GA1UE
AxMJbG9jYWxob3N0MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDFbNC
qnWSy
SkBPufp3yzhhZYEG7wPxfLaQY2vakm+LHiGmVT2M2SMFdCH4k3AlNnCn9KMg5
Ji9
mDPB0Bdh+Z1AhhZjyAcqGdVDUKnkbwKETURIgGyshtWS1YHAu9JQXlDnifSNq/
WB
w+2mJsXz89dMBSGTIDLxYvLXDS83upicSwIDAQABMA0GCSqGSIb3DQEBBQUA
A4GB
AFstUGWsMTNEiMKemWpBZFgzCa/SkNAkR/x7I3JT9LO1gtyLdflHNueIi3JWQFh
m
Z00OAUfKR6vkrsGda2ITnLFrsemA3uAi4pbGZxyUryBB3lojQRCp/nUEOOhJ3Cpg
zlozZXSWdsvDCCyXF3f8oSoLzVZ+RuI/4xKEor6DDHE6
-----END CERTIFICATE-----

Le code

Accueil.php

Page qui gère l'identification des utilisateurs déjà enregistrés. Deux champs (email et mot de passe) sont blindés.
Un bouton permet d'être redirigé vers la page «inscription.php » et un autre permet de s'identifier pour accéder à l'accueil du site. (cette même page « accueil.php »)

Inscription.php

Page qui permet d'inscrire de nouveaux utilisateurs au site.

Edit.php

Page qui permet à l'utilisateur déjà connecté de modifier son profil.

Protection.php

Contient le code de blindage des champs qui est appelé dans les autres pages.
On se sert des méthodes mysql_real_escape_string et htmlentities pour le blindage vu que magic_quotes_gpc est OFF sur le serveur Apache de la machine virtuelle.
Contient également les codes de chiffrement symétrique (librairie mcrypt) qui nous aurait servi à crypter les données avant de les envoyer dans la base de données et à les déchiffrer avant des les renvoyer à l'utilisateur.

La base de données

Table utilisateur

Champs : email, pseudo, nom, prénom, rue, numéro, code postal, ville, numéro de téléphone, numéro de fax.

Threat Modeling

Le document d'information

Etape de developpement : ver 1.0
Directeur : /
Participants : Balon-Perin Alexandre, Pham Quang Vinh, Rukundo Patrick
Relecteur : /
Localisation : /
Description : Dans le cadre du troisième projet du cours de Computer Security, il nous est demandé de réaliser un site PHP/MySQL sur server Apache qui soit le plus sécurisé possible.

Les scenarii d'utilisation

ID: 1
Description : Le site est hébergé en localhost sur un server Apache installé sur une machine virtuelle PepperMint OS. Le serveur communiquera avec le client via le protocole OpenSSL.

ID: 2
Description: La base de données du site est hébergée sur phpMyAdmin. Elle est protégée par un mot de passe.

Les dépendances externes

ID: 1
Description : la sécurité du site dépend du serveur de pages web. Nous utiliserons Apache 2 grâce auquel nous utiliserons le protocole OpenSSL.

ID: 2
Description: la sécurité de la base de données dépend du serveur sur laquelle elle est installée.

Les hypothèses d'implantation

ID: 1
Description: les communications chiffrées se feront selon le protocole OpenSSL

ID: 2
Description: les champs de formulaire faisant office de passerelle entre serveur web et la base de données seront blindés afin d'éviter toute injection.

Les notes de sécurité externes

ID: 1
Description: Néant.

Les notes de sécurité internes

ID: 1
Description: le serveur n'est pas protégé contre les injections SQL, c'est donc aux développeurs de s'assurer de blinder les champs.

ID: 2
Description: la base de données n'est pas sécurisée, il faudrait donc crypter les champs sensibles.

Les niveaux de confiance

ID: 1
Nom: passant
Description: personne qui n'étant pas inscrite au site ne peut voir que le formulaire de login ou le formulaire d'inscritpion.

ID: 2
Nom: utilisateur du site
Description: personne qui est inscrite sur le site et peut donc en voir toutes les pages.

ID: 3
Nom: HTTP user
Description: personne qui accède au site via HTTP.

ID: 4
Nom: HTTPS user
Description: personne qui accède au site via HTTPS.

ID: 5
Nom: webmaster/database administrator
Description: personne qui s'occupe d'écrire et maintenir le code du site/la base de données. Il aura créé un utilisateur à son nom.

Les points d'entrées

ID: 1
Nom: page de login
Description: Page internet à partir de laquel les utilisateurs peuvent se connecter pour accéder à l'application
Niveau de confiance : (2)utilisateurs du site, (5) webmaster, (1) passant,(3) HTTP user, (4) HTTPS user

ID: 2
Nom: fonction Login
Description: Compare les informations entrées par l'utilisateur avec les informations présentent dans la base de données et crée une nouvelle session si elles correspondent.
Niveau de confiance: (2)utilisateurs du site, (5) webmaster, (4)HTTPS user

ID: 3
Nom: fonction Register
Description: Crée un nouvel utilisateur.
Niveau de confiance: (1)passant, (4)HTTPS user

ID: 4
Nom: fonction Edit
Description: Permet à l'utilisateur de modifier ses informations de profile.
Niveau de confiance: (2)utilisateur du site, (5)webmaster, (4)HTTPS user

ID: 5
Nom: ports de communications du serveur web(HTTPS: 443)
Description: Le port que le serveur web écoute.
Niveau de confiance: (1)passant, (2) utilisateurs du site,(4)HTTPS user,(5)webmaster

Les actifs

ID: 1
Nom: données de login
Description: recevoir l'email et le mot de passe d'un user
Niveau de confiance: (2)utilisateur du site

ID:2
Nom: données personelles de l'utilisateur.
Description: les données personelles que l'utilisateur entre dans les formulaires, comme son adresse, son nom,etc.
Niveau de confiance: (2)utilisateur du site

ID:3
Nom: System
Description: actifs relatifs au système sous-jacent.

ID : 4
Nom: Disponibilité du site
Description : si le site web crash, les utilisateurs n'ont plus accès à leurs données personelles.
Niveau de confiance : (5) webmaster/database administrator.

Les menaces

ID: 1
Nom: L'adversaire accède sur le site sans entrer de login
STRIDE: E
Evincement connu: on empêche toute injection SQL. On écrit la base de données en français, ainsi le nom des champs est moins "évident" à retrouver.
Notes d'investigation: vérifier que les magic quotes sont bien activées sur Apache.
Points d'entrées: page de login.
Actifs: login de session

ID: 2
Nom: L'adversaire trouve le mot de passe d'un utilisateur.
STRIDE: E
Evincement connu: protection contre injection SQL, hashage des mots de passe.
Points d'entrées: page de login.
Actifs: login de session

ID: 3
Nom: L'adversaire récupère les informations en écoutant les ports de communication lors du login ou de l'inscription.
STRIDE: E
Evincement connu: encodage des données avant de passer dans le canal de communication, grâce à OpenSSL (cryptage asymétrique).
Points d'entrée: ports sur écoute du serveur web.
Actifs: page de login, page d'inscription.

ID: 4
Nom: L'adversaire arrive à pénétrer jusque dans la base données.
STRIDE: T et E
Evincement connu: crypter les informations contenues dans la base de données avec un cryptage symétrique.
Actifs: login de la bdd

ID: 5
Nom: L'adversaire se connecte sous le nom d'un autre utilisateur sans son mot de passe et veut changer ses données.
STRIDE: T I
Evincement connu: le formulaire d'edition ne remet pas automatiquement le mot de passe, il faut au moins rentrer l'ancien mot de passe pour pouvoir confirmer les changements.
Points d'entrée: le formulaire d'édition
Actif: le formulaire d'édition

ID: 6
Nom: L'adversaire se connecte sous le nom d'un autre utilisateur sans son mot de passe et veut le changer.
STRIDE: T I
Evincement connu: demander l'ancien mot de passe avant de pouvoir inscrire le nouveau.
Points d'entrée: le formulaire d'édition.
Actif: le formulaire d'édition

Diagramme de flux

contrôle de