SECURE SPHERE by EPITA

SECURE DEV

Introduction

Quentin GROSYEUX

2023 - BNP

Table des matières

Table des matières

Un mot sur moi

- EPITA, spécialité systèmes, réseau et sécurité
- Formateur pour SecureSphere

Généralités

Objectifs

Formation, grâce à des exemples concrets et réels, à la sécurité logicielle, à travers toutes les étapes d'un projet

Méthode :

- exposés avec des questions
- questions ouvertes, ateliers de réflexion
- travaux pratiques

Déroulement de la formation

- les supports présentés seront fournis en version électronique (PDF), mais strictement réservés à l'usage personnel des stagiaires qui ont suivi la formation (pas de diffusion ni de réutilisation)
- l'enregistrement audio ou vidéo est interdit

Déroulement de la formation

- horaires :
  - 9h30 - 10h15
    pause
  - 10h30 - 12h
    déjeuner
  - 13h - 15h
    pause
  - 15h15 - 17h30

Déroulement de la formation

- remise en cause potentielle de certaines habitudes
- aucun jugement de ma part et entre vous
- chacun a ses propres compétences, la sécurité peut être un nouveau domaine pour certains d'entre vous
- votre éventuelle faible connaissance en sécurité ne remet pas en cause vos autres compétences
- évitons les sarcasmes et dénonciations entre vous sur les exemples liés à vos produits !

Vous

- vos métiers ?
- votre expérience en sécurité informatique ?
- attentes particulières ?

Plan

1. introduction
2. conception
3. développement
4. recette, intégration, distribution et déploiement

Certification QUALIOPI de SecureSphere

- certification pour les organismes de formation
- pour vous : juste un quiz au début de la formation (maintenant) et à la fin, pour valider l'acquisition des compétences, et un petit formulaire d'enquête qualité à la fin

Exercices de mise en pratique

- application volontairement vulnérable mais (presque) sans tricher
- exemples parfois simplistes mais réalistes
- multiplication des fonctionnalités pour brosser tous les problèmes
- je ne suis pas un développeur Java ☺

Superbouchons

Exercice

Découvrir l'application et trouver les vulnérabilités manuellement

- VM Linux 64 bits avec XFCE (`liveuser`/`live`)
- JDK, Eclipse et outils utiles pour les TP
- depuis le répertoire `~/java_spring/` :
  - `./compile.sh`
  - `./run.sh`

S'assurer que le navigateur puisse accéder à Internet (carte réseau en mode NAT)

Table des matières

"Piratages" médiatisés

Exemples d'attaques ou de failles médiatisées ?

Virus / ver

- ver "Shamoon" (août 2012) : propagation et effacement du MBR (30 000 postes dans la société pétrolière Aramco)

Virus / ver

- ver "Shamoon" (août 2012) : propagation et effacement du MBR (30 000 postes dans la société pétrolière Aramco)

⇒ virus, ver, propagation, charge (*payload*), sabotage

Virus / ver

- ver "Shamoon" (août 2012) : propagation et effacement du MBR (30 000 postes dans la société pétrolière Aramco)

⇒ virus, ver, propagation, charge (*payload*), sabotage

- ver "Wannacry" (2017)

Virus / ver

- ver "Shamoon" (août 2012) : propagation et effacement du MBR (30 000 postes dans la société pétrolière Aramco)

⇒ virus, ver, propagation, charge (*payload*), sabotage

- ver "Wannacry" (2017)

⇒ rançongiciel (*ransomware*)

Virus / ver

- ver "Shamoon" (août 2012) : propagation et effacement du MBR (30 000 postes dans la société pétrolière Aramco)

⇒ virus, ver, propagation, charge (*payload*), sabotage

- ver "Wannacry" (2017)

⇒ rançongiciel (*ransomware*)

- ver "NotPetya" (2017)

Compromission de postes ou serveurs

- plus de 500 000 ordinateurs compromis par BlackShade en 2014 (40$ sur Internet)

Compromission de postes ou serveurs

- plus de 500 000 ordinateurs compromis par BlackShade en 2014 (40$ sur Internet)

⇒ Exploit kit, botnet, machines zombies

Compromission de postes ou serveurs

- plus de 500 000 ordinateurs compromis par BlackShade en 2014 (40$ sur Internet)

⇒ Exploit kit, botnet, machines zombies

- Mirai (2016)

Compromission de postes ou serveurs

- plus de 500 000 ordinateurs compromis par BlackShade en 2014 (40$ sur Internet)

⇒ Exploit kit, botnet, machines zombies

- Mirai (2016)

⇒ Mots de passe par défaut, IoT, déni de service (*denial of service*)

Compromission de postes ou serveurs

- plus de 500 000 ordinateurs compromis par BlackShade en 2014 (40$ sur Internet)

⇒ Exploit kit, botnet, machines zombies

- Mirai (2016)

⇒ Mots de passe par défaut, IoT, déni de service (*denial of service*)

- site Internet de Pathé (2016)

Compromission de postes ou serveurs

- plus de 500 000 ordinateurs compromis par BlackShade en 2014 (40$ sur Internet)

⇒ Exploit kit, botnet, machines zombies

- Mirai (2016)

⇒ Mots de passe par défaut, IoT, déni de service (*denial of service*)

- site Internet de Pathé (2016)

⇒ Point d'eau (*water holing*), rançongiciel

Compromission de postes ou serveurs

- plus de 500 000 ordinateurs compromis par BlackShade en 2014 (40$ sur Internet)

⇒ Exploit kit, botnet, machines zombies

- Mirai (2016)

⇒ Mots de passe par défaut, IoT, déni de service (*denial of service*)

- site Internet de Pathé (2016)

⇒ Point d'eau (*water holing*), rançongiciel

- 2 millions d'ordinateurs dans le monde avec le logiciel constructeur "Computrace" pouvant être détourné depuis Internet

Criminalité

- ebay (février/mars 2014) : noms de clients, mots de passe chiffrés, adresses mail, dates de naissance, adresses postales et numéros de téléphone de 145 millions de clients

Criminalité

- ebay (février/mars 2014) : noms de clients, mots de passe chiffrés, adresses mail, dates de naissance, adresses postales et numéros de téléphone de 145 millions de clients
- TV5 Monde (2015)

Criminalité

- ebay (février/mars 2014) : noms de clients, mots de passe chiffrés, adresses mail, dates de naissance, adresses postales et numéros de téléphone de 145 millions de clients
- TV5 Monde (2015)
- banque centrale du Bangladesh (2016)

Vidéos

Exemples :

- élévation locale de privilège sous Linux
- compromission d'un poste de travail *via* un fichier PDF malveillant

Types d'exploitation

- Physique :
- Locale :
- Distante :

Types d'exploitation

- Physique :
  - service d'authentification interactive (graphique ou texte), écran de veille, noyau (gestion USB, FireWire, pilote de système de fichiers, de protocole sans fil, etc.)
- Locale :
- Distante :

Types d'exploitation

- Physique :
  - service d'authentification interactive (graphique ou texte), écran de veille, noyau (gestion USB, FireWire, pilote de système de fichiers, de protocole sans fil, etc.)
- Locale :
  - service en administrateur, programme qui s'exécute automatiquement en administrateur, noyau (pilote accessible aux programmes, appels système)
- Distante :

Types d'exploitation

- Physique :
  - service d'authentification interactive (graphique ou texte), écran de veille, noyau (gestion USB, FireWire, pilote de système de fichiers, de protocole sans fil, etc.)
- Locale :
  - service en administrateur, programme qui s'exécute automatiquement en administrateur, noyau (pilote accessible aux programmes, appels système)
- Distante :
  - serveurs : service écoutant sur le réseau (application), noyau (pilote de protocole réseau), etc.

Types d'exploitation

- Physique :
  - service d'authentification interactive (graphique ou texte), écran de veille, noyau (gestion USB, FireWire, pilote de système de fichiers, de protocole sans fil, etc.)
- Locale :
  - service en administrateur, programme qui s'exécute automatiquement en administrateur, noyau (pilote accessible aux programmes, appels système)
- Distante :
  - serveurs : service écoutant sur le réseau (application), noyau (pilote de protocole réseau), etc.
  - postes de travail : idem serveur, toute application manipulant des fichiers reçus (suite bureautique, lecteurs multimédia, navigateur Internet, etc.)

Vocabulaire

Vulnérabilité (RFC 2828) :

0day :

Exploit :

Preuve de concept (PoC) :

Vocabulaire

Vulnérabilité (RFC 2828) : faille ou faiblesse qui pourrait être exploitée pour échapper à la politique de sécurité

0day :

Exploit :

Preuve de concept (PoC) :

Vocabulaire

Vulnérabilité (RFC 2828) : faille ou faiblesse qui pourrait être exploitée pour échapper à la politique de sécurité

0day : vulnérabilité dont le correctif de sécurité n'est pas disponible

Exploit :

Preuve de concept (PoC) :

Vocabulaire

Vulnérabilité (RFC 2828) : faille ou faiblesse qui pourrait être exploitée pour échapper à la politique de sécurité

0day : vulnérabilité dont le correctif de sécurité n'est pas disponible

Exploit : programme, script ou ligne de commande permettant d'exploiter réellement une vulnérabilité et d'exécuter une charge utile (*payload*)

Preuve de concept (PoC) :

Vocabulaire

Vulnérabilité (RFC 2828) : faille ou faiblesse qui pourrait être exploitée pour échapper à la politique de sécurité

0day : vulnérabilité dont le correctif de sécurité n'est pas disponible

Exploit : programme, script ou ligne de commande permettant d'exploiter réellement une vulnérabilité et d'exécuter une charge utile (*payload*)

Preuve de concept (PoC) : début d'exploit qui permet d'atteindre une vulnérabilité dans un environnement particulier et sans exécuter de charge utile complète

0days

Project Zero 0day "In the Wild" (2019-05-15)

`https://docs.google.com/spreadsheets/d/1lkNJ0uQwbeC1ZTRrxdtuPLCIl7mlUreoKfSIgajnSyY/view`

Conséquences d'une vulnérabilité

Conséquences d'une vulnérabilité

- élévation de privilèges (*Local Privileges Escalation*)
- exécution de code (*Local Code Execution*, *Remote Code Execution*) arbitraire
- exécution d'une action non prévue pour l'utilisateur
- arrêt inopiné (*Denial Of Service*)
- divulgation d'informations techniques ou métier (*Information Disclosure*, *Information Leak*)
- modification d'informations (*Tampering*)
- requête réseau vers une ressource interne (*Server-Side Request Forgery*)

Dénis de service

Rendre inaccessible un serveur/service :

- exploitation d'une erreur d'implémentation (*ping of death*)

⇒ persistant jusqu'au redémarrage du service ou de la machine

- épuisement temporaire de ressources

⇒ reprise du service à l'arrêt de l'attaque

  - asymétrie conceptuelle entre le serveur et les clients (*syn flood*, *sockstress*, requête applicative très gourmande, etc.)
  - charge trop importante : déni de service distribué (*distributed denial of service*)

Mises à jour

Correctifs de sécurité :

- 10 529 vulnérabilités enregistrées depuis le début de l'année 2017
- Chrome 61 (septembre 2017) : correction de 22 vulnérabilités
- Microsoft Patch Tuesday septembre 2017 : 38 correctifs pour 81 vulnérabilités
- Google (septembre 2017) : 83 vulnérabilités dans Android et drivers Qualcomm
- Oracle (juillet 2017) : 310 vulnérabilités sur une cinquantaine de produits
- SAP security patch day (septembre 2017) : 23 security notes

CVE

- Common Vulnerabilities and Exposure
- géré par le MITRE
- base de données publique d'identifiants des vulnérabilités
- CVE-AAAA-NNNNN
- NVD : base de données publique des vulnérabilités

CVE

Exercice

Lister les 5 dernières CVE, déterminer les informations présentes dans une entrée CVE et les catégories de vulnérabilités

`cvedetails.com`

CVSS

- Common Vulnerability Scoring System
- version 3.1 publiée en 2019
- décrit les caractéristiques et les niveaux de criticité
- génère un score de criticité

Métriques CVSS

Formulaire de calcul du score CVSS :

- métrique de base (intrinsèque) :
  - exploitabilité
  - impact
- métrique temporelle
- métrique environnementale

Score CVSS

Score de criticité :

- 0 : aucun
- 0.1 - 3.9 : faible
- 4 - 6.9 : modéré
- 7 - 8.9 : élevé
- 9 - 10 : critique

Score CVSS

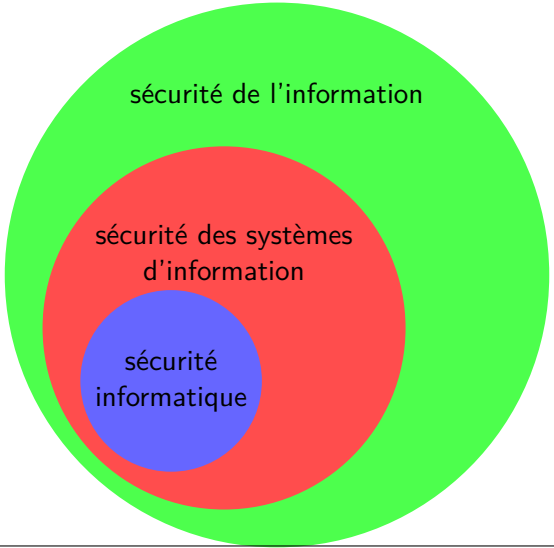
Exercice

Grâce au formulaire sur le site de CVSS, déterminer le score CVSS d'une faille fictive :

- sur un site Web publiquement accessible, permettant d'exécuter une commande sous l'identité du compte non privilégié du service Web
- sur un site Web interne à une entreprise, permettant de récupérer l'ensemble des données de la base SQL associée au site

Table des matières

Sécurité des systèmes d'information



Idées pré-conçues à éviter

- Je sécuriserai plus tard
- Ce n'est pas mon travail, il y a une équipe sécurité pour cela
- Je n'ai pas la compétence
- C'est trop chronophage
- Mon application est interne, pas besoin de la sécuriser

- Mon application ne se fera jamais pirater
- Inutile d'être paranoïaque

Idées pré-conçues à éviter

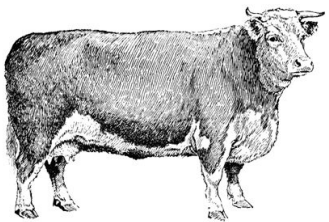
- Je sécuriserai plus tard
- Ce n'est pas mon travail, il y a une équipe sécurité pour cela
- Je n'ai pas la compétence
- C'est trop chronophage
- Mon application est interne, pas besoin de la sécuriser
  - Prestataire externe, usage de code tiers (bibliothèques, conteneurs), etc.
- Mon application ne se fera jamais pirater
- Inutile d'être paranoïaque

Coût des vulnérabilités

Selon le National Institute of Standards and Technology (NIST) en 2002, les corrections de code appliquées après la mise sur le marché peuvent coûter 30 fois plus cher que les corrections effectuées lors de la phase de conception.

Coût des vulnérabilités

No comments, no documentation but 20 tickets



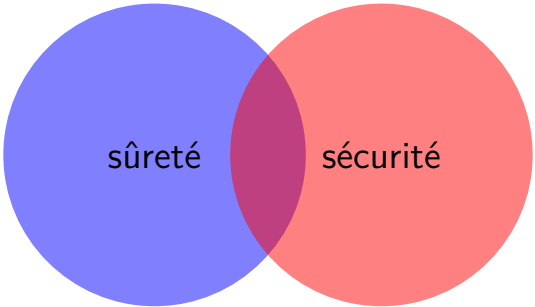
The Guy Who Wrote This Is Gone

It's running everywhere

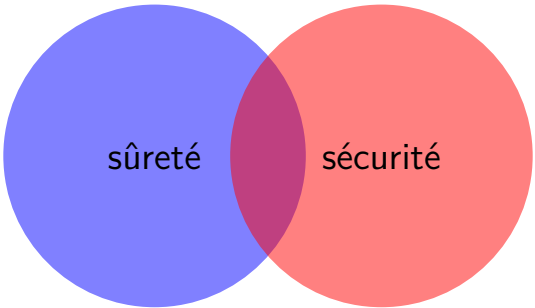
O RLY?

FML

Sécurité vs sûreté

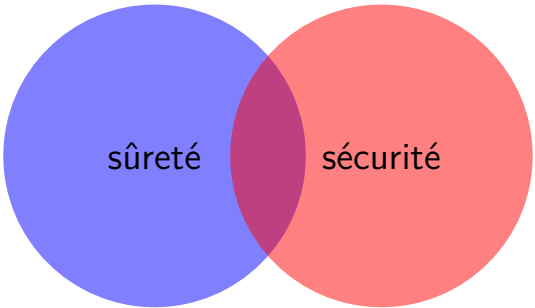


Sécurité vs sûreté



tolérance à
une panne

Sécurité vs sûreté



tolérance à une panne

robustesse face à une attaque

Surface d'attaque/d'exposition

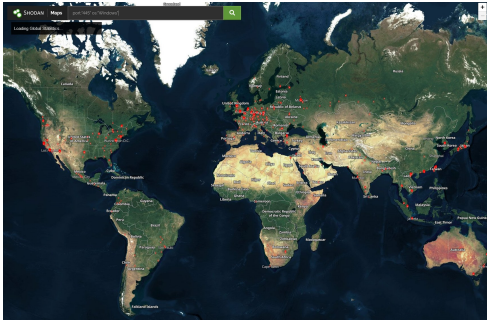
Surface d'attaque/d'exposition

- toute fonctionnalité et tout composant qui manipule des données provenant de l'extérieur
- toutes les lignes de code applicatif qui gèrent des données de l'utilisateur
- tout le code qu'un attaquant peut atteindre pour trouver et exploiter une vulnérabilité

Nécessaire d'évaluer et de réduire la surface d'attaque

Surface d'attaque des serveurs Windows exposés sur Internet

Avril 2022, CVE-2022-26809 "Remote Procedure Call Runtime Remote Code Execution Vulnerability"
Exposition du port 445 sur Internet



(Source : `https://twitter.com/UK_Daniel_Card/status/1514129322667352064`)

Surface d'attaque d'un navigateur Web

Surface d'attaque d'un navigateur Web

- protocole HTTP (en-tête, compression, fragmentation, etc.)
- protocole TLS (format ASN.1, certificats, algorithmes cryptographiques)
- analyse du code HTML et rendu
- manipulation du DOM
- moteur Javascript et CSS
- gestion des fichiers (cookies, historique, formulaires, mots de passe, cache, etc.)
- interface graphique (barre d'URL, menus, presse-papier, page de configuration)
- bibliothèques de format d'images et de son
- gestion des plugins et code de chaque plugin : PDF, office, flash, java, silverlight, Active-X, etc.
- ...

Défense en profondeur

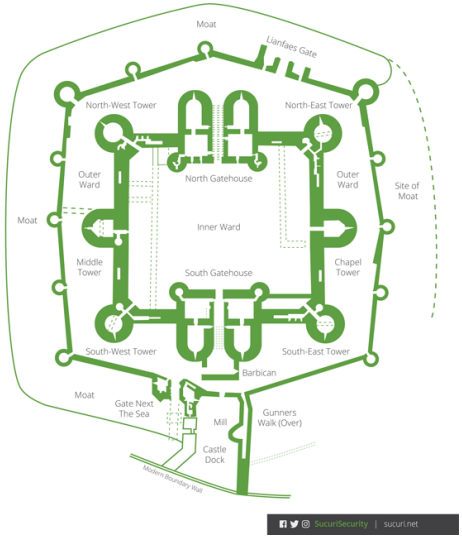
Défense en profondeur

Approche de construction des mécanismes de défense :

- plusieurs couches indépendantes de mesures de sécurité
- protection assurée même si une mesure de sécurité est contournée ou exploitée
- différents types de mesures de sécurité
- complexifie et ralentit l'attaque, facilite la détection des intrusions

Défense en profondeur

Défense en profondeur



Autres principes généraux

- moindre privilège / séparation des privilèges

Autres principes généraux

- moindre privilège / séparation des privilèges
- isolation

Autres principes généraux

- moindre privilège / séparation des privilèges
- isolation
- KISS

KISS

Autres principes généraux

- moindre privilège / séparation des privilèges
- isolation
- KISS : keep it simple, stupid

Table des matières

Résumé de la démarche d'analyse de risque

But

Déterminer les risques pesant sur l'application pour ne mettre en place que les mesures de sécurité pertinentes

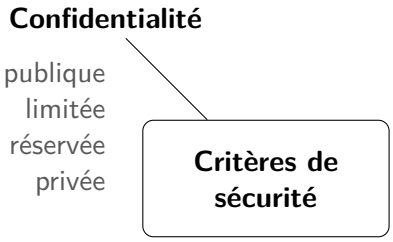
Étapes :

1. déterminer les éléments importants à protéger (biens essentiels / valeurs métier, biens support)
2. déterminer les besoins de sécurité des biens

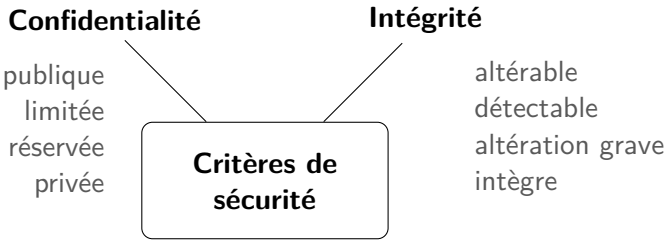
Besoins de sécurité

Critères de sécurité

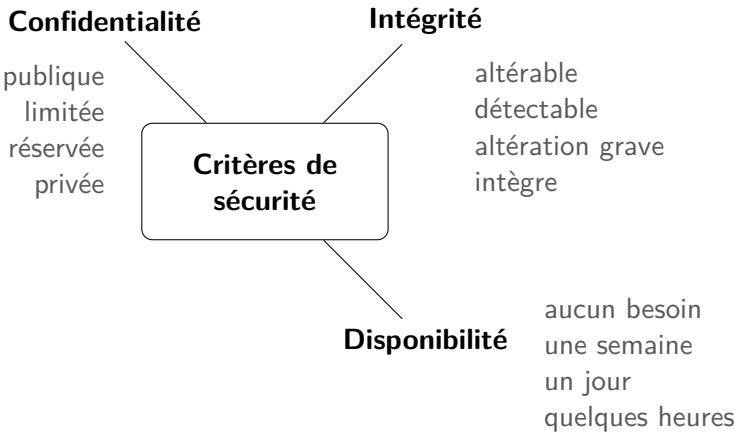
Besoins de sécurité



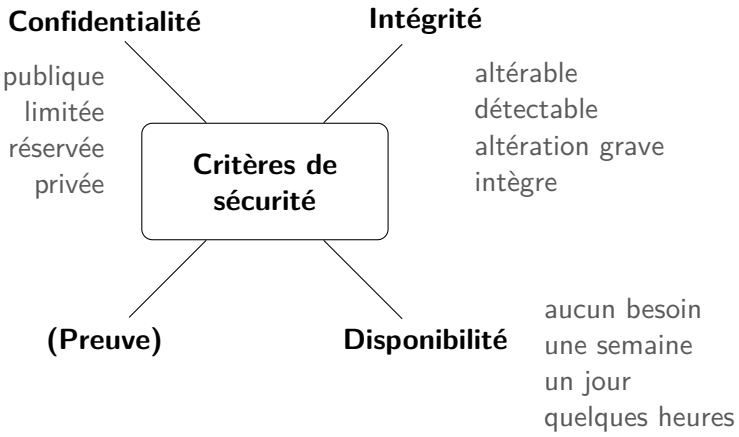
Besoins de sécurité



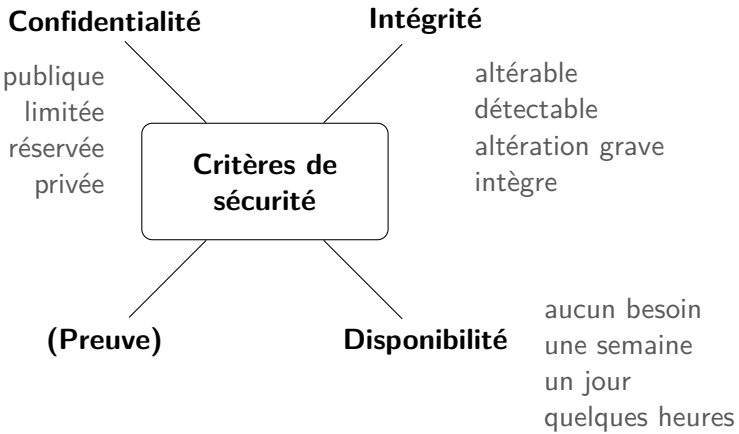
Besoins de sécurité



Besoins de sécurité



Besoins de sécurité



Échelle simplifiée : besoin important, besoin notable, aucun besoin

Résumé de la démarche d'analyse de risque

But

Déterminer les risques pesant sur l'application pour ne mettre en place que les mesures de sécurité pertinentes

Étapes :

1. déterminer les éléments importants à protéger (biens essentiels / valeurs métier, biens support)
2. déterminer les besoins de sécurité des biens (disponibilité, intégrité, confidentialité) et les évènements redoutés
3. déterminer contre qui protéger ces éléments et leurs objectifs

Utilisateurs légitimes

Est-il envisageable de penser que tous les utilisateurs légitimes d'un système sont bienveillants ?

Menace interne

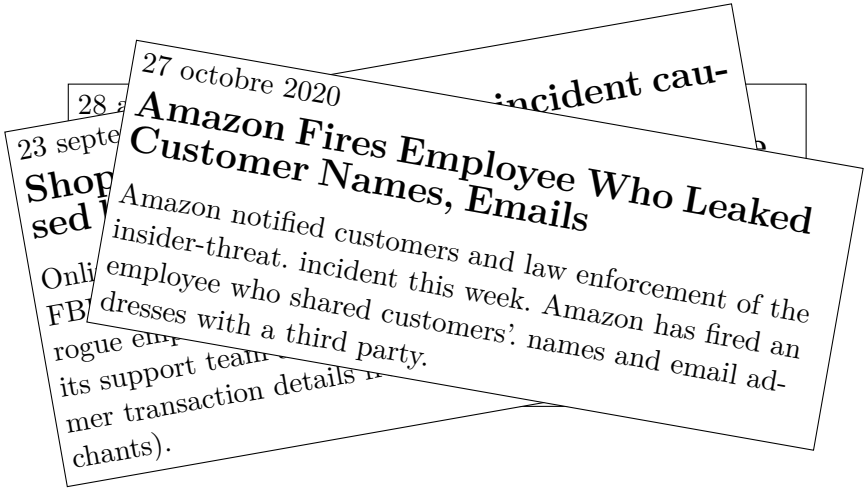
28 août 2020

Un salarié de Tesla refuse 1 million de dollars d'un pirate en échange d'informations

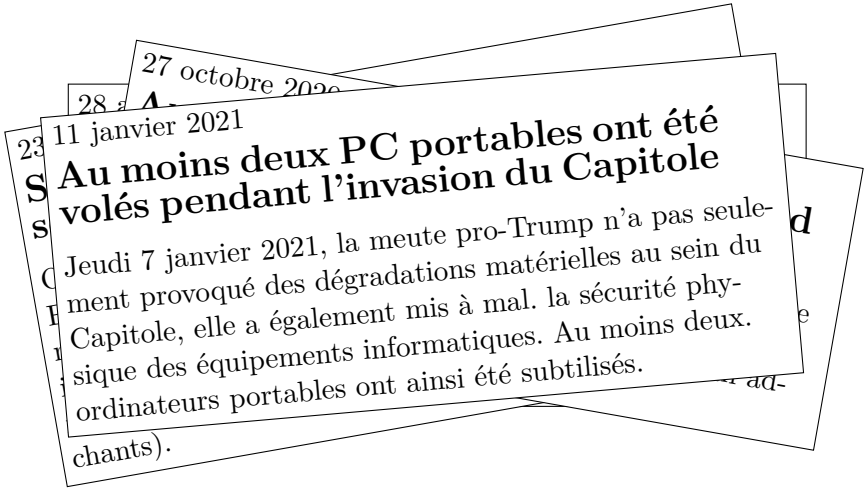
Le salarié de Tesla aurat été contacté par un cyberpirate russe qui voulait faire chanter l'entreprise. L'employé a laissé croire au pirate qu'il entrait dans son stratagème avant de le dénoncer à l'entreprise.

Menace interne

Menace interne



Menace interne



Résumé de la démarche d'analyse de risque

But

Déterminer les risques pesant sur l'application pour ne mettre en place que les mesures de sécurité pertinentes

Étapes :

1. déterminer les éléments importants à protéger (biens essentiels / valeurs métier, biens support)
2. déterminer les besoins de sécurité des biens (disponibilité, intégrité, confidentialité) et les évènements redoutés
3. déterminer contre qui protéger ces éléments et leurs objectifs
4. déterminer les scénarios stratégiques et scénarios opérationnels
5. déterminer les protections à mettre en œuvre

Démarche d'analyse de risque

- à effectuer au début du projet
- à mettre à jour à chaque changement structurel
- méthodes formalisées : EBIOS, Octave, Mehari, etc.
- version Agile et DevOps : *rapid risk assessment*

Contraintes de la sécurité

Security bugs

Linus Torvalds, 2008 : "*I don't think some spectacular security hole should be glorified or cared about as being any more "special" than a random spectacular crash due to bad locking*".

⇒ Linus Torvalds corrige des vulnérabilités en silence dans le noyau Linux, sans les marquer comme des correctifs de sécurité.

Open source

Les logiciels *open source* sont-ils plus sécurisés que les logiciels *close source* ?

Open source

Mozilla Foundation Security Advisory 2008-21 :

```
- name = js_AtomToPrintableString(cx, JSID_TO_ATOM(id));
+ name = js_ValueToPrintableString(cx, ID_TO_VALUE(id));
```

Table des matières

Phase de conception

La sécurité doit être intégrée dès le début du projet (conception)

Solutions

- ✓ réaliser une analyse de risque en amont
- ✓ se renseigner sur les utilisations des composants tiers
- ✓ nommer un responsable sécurité pour le projet
- ✓ faire appel à des personnes connaissant le domaine
- ✓ exprimer clairement les besoins (fonctionnels et techniques) dans le cahier des charges, se renseigner sur les bonnes pratiques
- ✓ choisir les solutions proposées également sous l'angle de la sécurité

Secure by design

Exemples de critères de comparaison des projets open source

Exemples de critères de comparaison des projets open source

- activité : dernier commit de moins de 3 mois, plusieurs contributeurs principaux
- dépendances : versions récentes et sans vulnérabilité connue
- qualité de code : nombre et état des tests unitaires
- vulnérabilités : délai de correction des CVE précédentes

Gestion de projet

Tout ce qui n'est pas écrit ne sera pas fait.

Tout ce qui est mal spécifié sera mal implémenté.

Phase de contractualisation

Si le développement est externalisé, il faut intégrer des clauses liées au développement dans le contrat

Phase de contractualisation

Si le développement est externalisé, il faut intégrer des clauses liées au développement dans le contrat

Solutions

- ✓ engagement du prestataire sur la formation en sécurité des développeurs
- ✓ engagement du prestataire à faire son possible pour minimiser les vulnérabilités relatives au développement
- ✓ écriture par le prestataire d'une documentation sur les mécanismes mis en œuvre pour éviter les vulnérabilités courantes et pour implémenter les mesures de sécurité (authentification, autorisation, etc.)
- ✓ propriété du code source (réversibilité de la TMA) en cas de faillite du prestataire ou de rachat par une autre société

Phase de contractualisation

Si le développement est externalisé, il faut intégrer des clauses liées à l'audit dans le contrat

Phase de contractualisation

Si le développement est externalisé, il faut intégrer des clauses liées à l'audit dans le contrat

Solutions

- ✓ audit du prestataire (SI, processus de développement, etc.)
- ✓ audit de la prestation (code source, etc.)
- ✓ intégration du plan d'action suite à l'audit

Phase de contractualisation

Si le développement est externalisé, il faut intégrer des clauses de maintien en condition de sécurité (MCS) dans le contrat

Phase de contractualisation

Si le développement est externalisé, il faut intégrer des clauses de maintien en condition de sécurité (MCS) dans le contrat

Solutions

- ✓ définition d'un processus de gestion et de facturation des correctifs de sécurité
- ✓ engagement du prestataire à rechercher partout dans le projet la présence des vulnérabilités qui lui sont remontées
- ✓ engagement du prestataire à diffuser des mises à jour si des composants tiers utilisés sont mis à jour
- ✓ conditions d'évolution de l'environnement (migration du système d'exploitation, changement de configuration du système, etc.)

Phase de développement

Solutions

- ✓ les développeurs doivent être formés à la sécurité

Phase de développement

Solutions

- ✓ les développeurs doivent être formés à la sécurité
- ✓ un standard de code doit être suivi

Phase de développement

Solutions

- ✓ les développeurs doivent être formés à la sécurité
- ✓ un standard de code doit être suivi
- ✓ des guides de développement sécurisé doivent être suivis

Phase de développement

Solutions

- ✓ les développeurs doivent être formés à la sécurité
- ✓ un standard de code doit être suivi
- ✓ des guides de développement sécurisé doivent être suivis
- ✓ des outils particuliers doivent être utilisés pour éviter les vulnérabilités classiques (analyse automatique de code, options de compilation, etc.)

Phase de développement

Solutions

- ✓ les développeurs doivent être formés à la sécurité
- ✓ un standard de code doit être suivi
- ✓ des guides de développement sécurisé doivent être suivis
- ✓ des outils particuliers doivent être utilisés pour éviter les vulnérabilités classiques (analyse automatique de code, options de compilation, etc.)
- ✓ des tests unitaires relatifs à la sécurité doivent être ajoutés

Phase de recette

Types de recette :

- recette fonctionnelle = s'assurer que l'application fait ce que l'on veut
- recette sécurité = s'assurer que l'application ne fait pas ce que l'on ne veut pas

Phase de recette

Solutions

- ✓ obtenir de la part des développeurs :
  - les détails de l'implémentation des mécanismes de sécurité (stockage des mots de passe, contrôle d'accès, etc.)
  - la gestion des cas particuliers et des entrées malveillantes
- ✓ analyser la facilité d'emploi et la confiance envers la configuration
- ✓ déterminer la surface d'attaque
- ✓ réaliser des tests techniques (tests d'intrusion) et faire éventuellement effectuer un audit (revue) de code source
- ✓ auditer les configurations des composants (socle système, services, applicatifs, etc.)

Autres phases

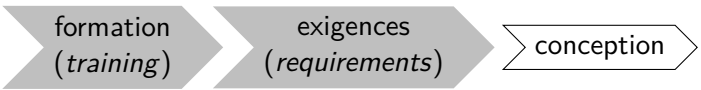
Solutions

- ✓ déploiement : guide d'installation et de configuration (suppression du mode debug, réactivation de certaines fonctions désactivées, etc.)
- ✓ intégration : gestion des droits, etc.
- ✓ exploitation : mise à jour, analyse des journaux, audits réguliers

Microsoft Security Development Lifecycle (1/4)

Détails sur `http://www.microsoft.com/sdl`.
Intégration depuis 2004 de la sécurité dans les développements Microsoft : 7 phases.

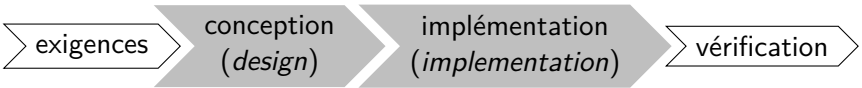
Microsoft Security Development Lifecycle (2/4)



- formation
- sensibilisation
- organisation

- jalons et livrables
- besoins de sécurité
- bugtraq et échelles
- analyse des risques

Microsoft Security Development Lifecycle (3/4)



Microsoft Security Development Lifecycle (4/4)

Fuzzer en open source (OneFuzz, 2020)

Microsoft Security Development Lifecycle

Exercice

Récupérer sur le site de Microsoft la documentation simplifiée sur SDL

Méthode Agile

Exercice

Récupérer le PDF de la version temporaire de l'intégration de la SSI dans une démarche Agile sur le site de l'ANSSI.

Contexte

Problématique

- la sécurité logicielle est un domaine parmi d'autres dans la sécurité (matérielle, physique, système, réseau, organisationnelle, etc.) mais **centrale** :
  - le but d'un serveur est de fournir un service
  - un poste client doit offrir des logiciels
- il est difficile et coûteux de sécuriser *a posteriori* une application et une architecture
- des dizaines de vulnérabilités sont découvertes chaque jour (aussi bien dans des logiciels propriétaires que libres)

Table des matières

CWE Top 25 Most Dangerous Software Errors 2021 (1/2)

- Out-of-bounds Write
- Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- Out-of-bounds Read
- Improper Input Validation
- Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
- Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
- Use After Free
- Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
- Cross-Site Request Forgery (CSRF)
- Unrestricted Upload of File with Dangerous Type
- Missing Authentication for Critical Function
- Integer Overflow or Wraparound

CWE Top 25 Most Dangerous Software Errors 2021 (2/2)

- Deserialization of Untrusted Data
- Improper Authentication
- NULL Pointer Dereference
- Use of Hard-coded Credentials
- Improper Restriction of Operations within the Bounds of a Memory Buffer
- Missing Authorization
- Incorrect Default Permissions
- Exposure of Sensitive Information to an Unauthorized Actor
- Insufficiently Protected Credentials
- Incorrect Permission Assignment for Critical Resource
- Improper Restriction of XML External Entity Reference
- Server-Side Request Forgery (SSRF)
- Improper Neutralization of Special Elements used in a Command ('Command Injection')

Table des matières

OWASP Top Ten 2021

Spécifique aux applications Web :

- A01:2021-Broken Access Control
- A02:2021-Cryptographic Failures
- A03:2021-Injection
- A04:2021-Insecure Design
- A05:2021-Security Misconfiguration
- A06:2021-Vulnerable and Outdated Components
- A07:2021-Identification and Authentication Failures
- A08:2021-Software and Data Integrity Failures
- A09:2021-Security Logging and Monitoring Failures
- A10:2021-Server-Side Request Forgery

OWASP API Security Top 10 2019

Spécifique aux API HTTP :

- API1 - Broken Object Level Authorization
- API2 - Broken User Authentication
- API3 - Excessive Data Exposure
- API4 - Lack of Resources & Rate Limiting
- API5 - Broken Function Level Authorization
- API6 - Mass Assignment
- API7 - Security Misconfiguration
- API8 - Injection
- API9 - Improper Assets Management
- API10 - Insufficient Logging & Monitoring