Sécurité informatique et confidentialité

virtualisation, conteneurs et exploitation de faille

Mathias RAMPARISON - prenom.nom@grenoble-inp.fr

2025

Outline

Actualité de la semaine...

- Sept. 29, 2025: Notepad++ DLL Hijacking Vulnerability Let Attackers **Execute Malicious Code**
- Sept. 27, 2025: PlugX and Bookworm Malware Attacks Target Asian Telecom and ASEAN Networks
- Sept. 27, 2025: Apache Airflow Vulnerability Exposes Sensitive Details to Read-Only Users
- Sept. 25, 2025: Cisco ASA **Zero-Day** Duo Under Attack; CISA Triggers Emergency Mitigation Directive
- Sept. 24, 2025: Hackers Exploit Pandoc **CVE-2025-51591** to Target AWS IMDS and Steal EC2 IAM Credentials

Autres : 100.000 euros par an: comment le salaire du responsable de sécurité informatique a flambé et Le gouvernement mobilise 250 millions d'euros pour doper le secteur de la cybersécurité

RGPD (2016)

- Révolution au niveau des engagements sécurités de toutes les entreprises manipulant des données d'utilisateurs européens depuis le 28 mai 2018
- Privacy by design
- Point important : "Sécurité par défaut"/Security by default... la sécurité DOIT être assurée (contrôle d'accès, prévention contre les failles, action en cas d'incident, PSSI, ...)

voir également Loi n° 2014-038 – Sur la protection des données à caractère personnel sur digital.gov.mg et Loi n°2014-006 sur la lutte contre la cybercriminalité

Outline

Rappel : Avertissement - La loi Article 323-1

- Le fait d'**accéder** ou de se maintenir, **frauduleusement**, dans tout ou partie d'un système de traitement automatisé de données est puni de **2 ans d'emprisonnement** et de **60.000 € d'amende**.

Rappel : Avertissement - La loi Article 323-1

- Le fait d'**accéder** ou de se maintenir, **frauduleusement**, dans tout ou partie d'un système de traitement automatisé de données est puni de **2 ans d'emprisonnement** et de **60.000 € d'amende**.
- Lorsqu'il en est résulté soit la **suppression** ou la **modification** de données contenues dans le système, soit une altération du fonctionnement de ce système, la peine est de **3 ans** d'emprisonnement et de **100.000 €** d'amende.

Rappel : Avertissement - La loi Article 323-1

- Le fait d'**accéder** ou de se maintenir, **frauduleusement**, dans tout ou partie d'un système de traitement automatisé de données est puni de **2 ans d'emprisonnement** et de **60.000 € d'amende**.
- Lorsqu'il en est résulté soit la **suppression** ou la **modification** de données contenues dans le système, soit une altération du fonctionnement de ce système, la peine est de **3 ans** d'emprisonnement et de **100.000 €** d'amende.
- Lorsque les infractions prévues aux deux premiers alinéas ont été commises à l'encontre d'un système de traitement automatisé de données à caractère personnel **mis en œuvre par l'Etat**, la peine est portée à cinq ans d'emprisonnement et à 150 000 € d'amende.

Avertissement - La loi Article 323-3

- Le fait d'**introduire frauduleusement** des données dans un système de traitement automatisé, d'**extraire, de détenir, de reproduire, de transmettre, de supprimer ou de modifier frauduleusement** les données qu'il contient est puni de **cinq ans** d'emprisonnement et de **150.000 €** d'amende.

Avertissement - La loi Article 323-3

- Le fait d'**introduire frauduleusement** des données dans un système de traitement automatisé, d'**extraire, de détenir, de reproduire, de transmettre, de supprimer ou de modifier frauduleusement** les données qu'il contient est puni de **cinq ans** d'emprisonnement et de **150.000 €** d'amende.
- Lorsque cette infraction a été commise à l'encontre d'un système de traitement automatisé de données à caractère personnel mis **en œuvre par l'Etat**, la peine est portée à **7 ans** d'emprisonnement et à **300.000 € d'amende.**

Terrains de jeux autorisés

- en local. . . ce que l'on va voir . . .
- Challenges de sécurité en ligne
  - **NewbieContest**
  - Zenk security
  - Root me
  - . . .

Exercice : résoudre **wargame 1 de newbiecontest**

Outline

Etude et compréhension des mécanismes 1

1. Quel est le service ou le programme compromis, quel est le type de compromission
2. Expliquer la vulnérabilité, décrire le mécanisme permettant de l'exploiter.
3. Cette faille concerne-t-elle des machines clientes ou des machines serveurs ?
4. Décrire une architecture typique du système d'information qui pourrait être impliquée dans l'exploitation de ces failles :

- les services mis en oeuvre,
- les machines concernées (clients et serveurs),
- les équipements réseaux,
- les réseaux d'interconnexion.

Etude et compréhension des mécanismes 2

5. Indiquer les mesures pour :
   - limiter l'impact de l'exploitation de ces failles ?
   - empêcher qu'elles ne puissent être exploitées ?
6. Référencer parmi les bonnes pratiques, celles qu'il faudrait utiliser pour limiter cette menace.
7. S'il s'agit d'une faille issue d'un développement, qu'est ce qu'il aurait fallu mettre en place dans les équipes de développement pour limiter l'apparition d'une telle faille.

Mise en oeuvre d'une expérimentation

Mise en oeuvre d'une faille

Mise en oeuvre d'une expérimentation

Mise en oeuvre d'une faille

-> mise en danger de votre matériel:

- droits root
- accès au système

Mise en oeuvre d'une expérimentation

Mise en oeuvre d'une faille

-> mise en danger de votre matériel:

- droits root
- accès au système

Comment faire?

-> Isoler :

- Machine dédiée jetable : pas vraiment réaliste

`Attention toutefois aux portes ouvertes`

Mise en oeuvre d'une expérimentation

Mise en oeuvre d'une faille

-> mise en danger de votre matériel:

- droits root
- accès au système

Comment faire?

-> Isoler :

- Machine dédiée jetable : pas vraiment réaliste
- Machine virtuelle (Vmware, virtualbox, . . .)

`Attention toutefois aux portes ouvertes`

Mise en oeuvre d'une expérimentation

Mise en oeuvre d'une faille

-> mise en danger de votre matériel:

- droits root
- accès au système

Comment faire?

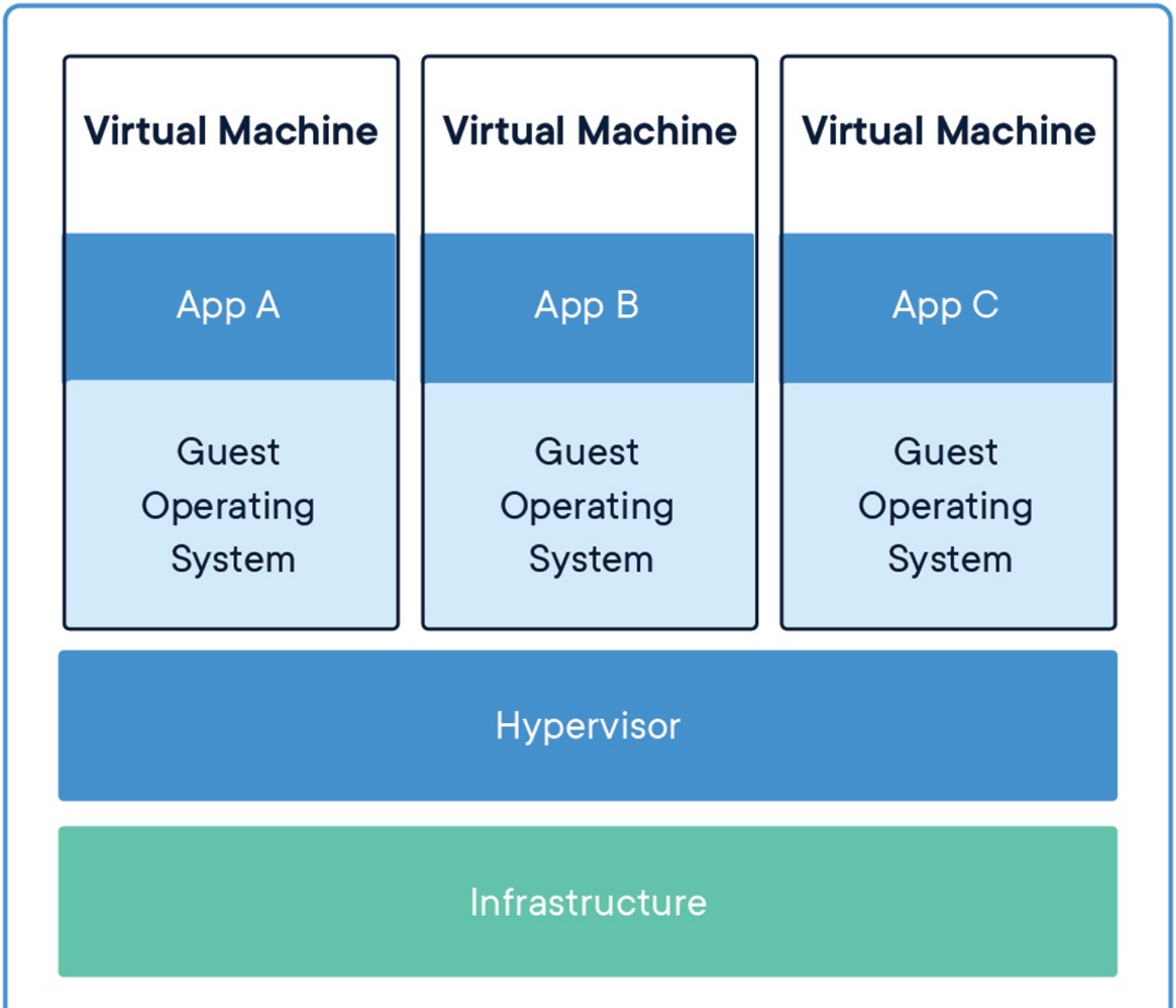
-> Isoler :

- Machine dédiée jetable : pas vraiment réaliste
- Machine virtuelle (Vmware, virtualbox, ...)
- Conteneur (LXC, Docker, ...)

`Attention toutefois aux portes ouvertes`

Outline

Concept de la virtualisation



Machines virtuelles

- Mise en place lourde :
  - Mise en place d'un autre système complet (hardware, OS, . . .)
- Isolation plus grande qu'avec un container
- Utiliser virtualbox
  - Utiliser vagrant (https://www.vagrantup.com/) et un **Vagrantfile**
- Interêts (autres que la *sécurité*) :
  - $$$
  - Stockage matériel
  - Sauvegardes
  - Ressources flexibles

Virtual Box et image à télécharger

- Télécharger et installer VirtualBox
  https://www.virtualbox.org/wiki/Downloads
- Télécharger une distribution légère, par exemple Puppy Linux en 32bits ou 64bits
- Créer une machine virtuelle avec cette distribution dans VirtualBox et :
  - aller dans `Applications -> Setup -> Puppy Packet Manager` et rechercher et installer et lancer par exemple `bastet` (normalement en version 0.43)
  - ouvrir *Puppy Installer* depuis *Setup* dans le menu principal, selectionner *frugal install* (the system files are stored in a small number of files in one directory, rather than spread out over a full partition)
  - pour aller plus loin, on peut mettre en place par exemple un dossier partagé entre l'host (votre machine) et le guest (la machine virtuelle)
  - exporter la machine virtuelle `File -> Export Appliance`

Exemple Vagrantfile

```
Vagrant.configure("2") do |config|
  config.vm.box = "debian/jessie64"
  config.vm.network "forwarded_port", guest:80,host: 13080
  config.vm.provision "shell", inline: <<-EOF
      sudo apt-get --assume-yes install apache2
  EOF
end
```

```
vagrant -h
vagrant up
vagrant port
vagrant ssh
vagrant destroy
```

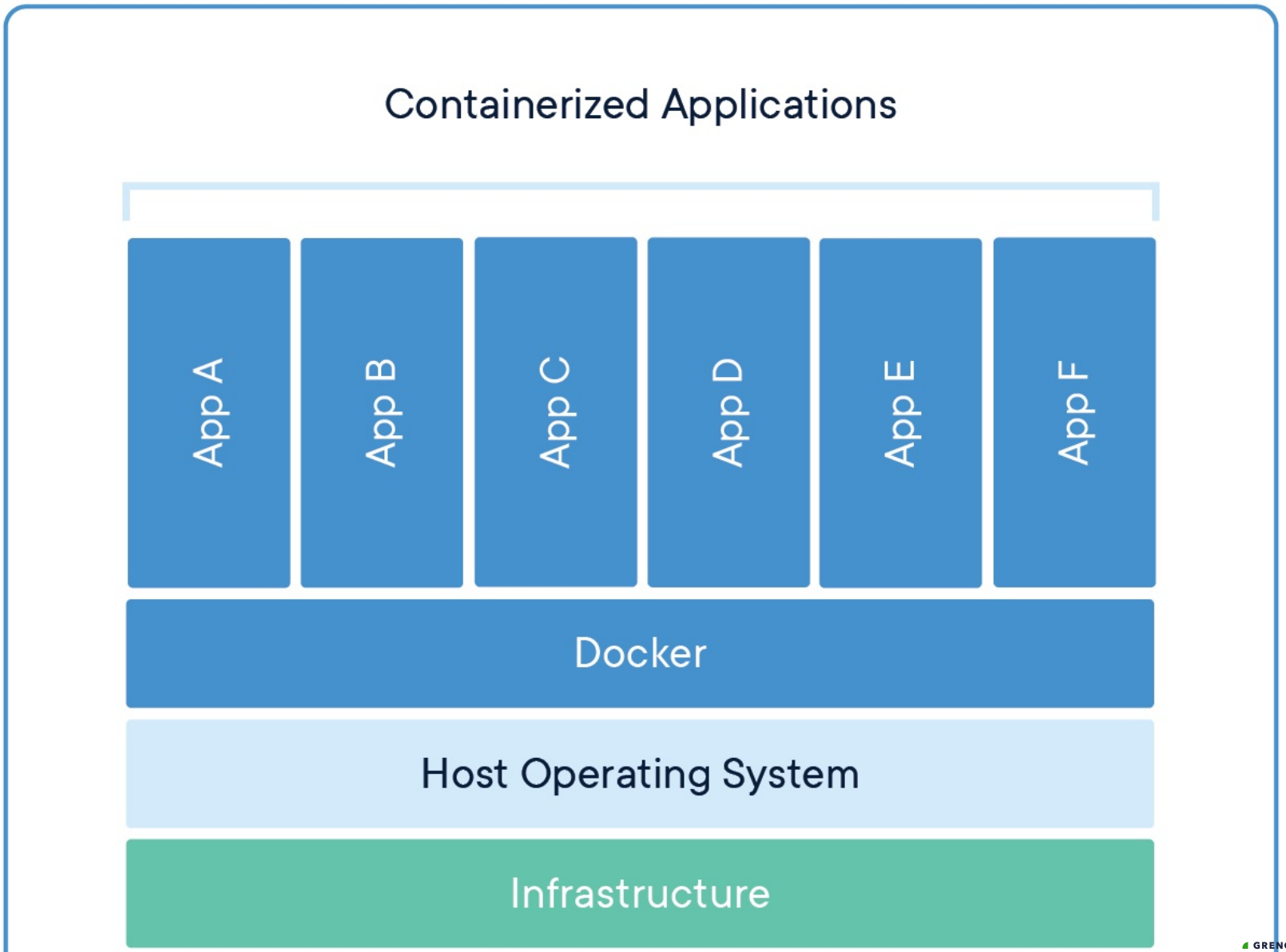
Remarque : un service ssh redirigé sur 2222 est automatiquement installé

Machines multiples (architecture plus réaliste) - Exemple

```
Vagrant.configure("2") do |config|
  config.vm.define "web" do |web|
    web.vm.box = "debian/strech64"
    web.vm.provision "file", source: "index.html",
      destination: "/tmp/index.html"
    web.vm.provision "shell", path: "apacheSetup.sh"
    web.vm.network "private_network", ip: "192.168.2.1",
      virtualbox__intnet: true
    web.vm.network "forwarded_port", guest:80,host: 13080
  end
  config.vm.define "db" do |db|
    db.vm.box = "debian/strech64" ...
    db.vm.network "private_network", ip: "192.168.2.2",...
  end
end
```

Outline

Principe d'un conteneur



Docker (illustrer des failles locales)

- Repose sur le système existant...
- Isole les processus grâce à cgroups, namespace, à chroot et le firewall de linux
- **Description dans une DockerFile**
- Pour OSX tourne au sein d'une machine virtuelle très légère
- **images** : stockage par couche - environnement système, applications, configurations, ...
  - système
  - connectivité
  - stockage persistant (à utiliser avec précaution dans notre cas!!!)
- **conteneur** : instance de l'image en exécution

Docker: environment setup for Ensimag computers

To start Docker, run

```
. <(curl 'https://viardots.gricad-pages.univ-grenoble-alpes.fr
# the script automatically launches a Docker environment
# within a VirtualBox virtual machine
```

Only the terminal in which you ran the previous command has the environment variables necessary for running Docker. If you open a new terminal, in order for the environment variables to be correctly set, use

```
. <(curl 'https://viardots.gricad-pages.univ-grenoble-alpes.fr
```

To stop the virtual machine at the end of the practical session:

```
. <(curl 'https://viardots.gricad-pages.univ-grenoble-alpes.fr
```

Alternativement, `podman` fonctionne sur les machines de l'Ensimag avec la même syntaxe

Commande et option

```
docker ps -a # liste les container
docker images # liste les images
docker run depot [cde] # lance le container avec éventuellemen
docker stop nomContainer # arrête un container démarré
docker start nomContainer # démarre
docker exec nomContainer cmd # Exécute une commande sur un co
```

Des options :

```
--rm : supprime le container une fois terminé
--i : interactif, -t : lier le tty à la machine
-p 6080:80 lier le port 80 du container au port 6080 local
-v /dir/local:/dir/container rendre accessible /dir/local
-e variable=nomVariable
--name nomContainer
-d detach mode (runs the container in background)
```

Docker : installation simple de container

On va récupérer l'image docker
https://hub.docker.com/r/pengbai/docker-supermario/

```
# Recherche dans les images disponibles
docker search docker-supermario
# Récupération de l'image
docker pull pengbai/docker-supermario
# On vérifie que l'on a bien récupéré l'image
docker images
# On lance le container # avec 8080:8080 ordis Ensimag
docker run -d -p 8080:8080 pengbai/docker-supermario
# On vérifie si le container est lancé
docker ps
```

On se rend avec son navigateur favori ici

```
# On arrête le container avec
docker stop ID_obtenu_au_dessus
```

Le fichier Dockerfile

Référence : https://docs.docker.com/engine/reference/builder/ Exemple : https://hub.docker.com/r/phocean/msf/~/dockerfile/

docker build -t nomImageCree .

```
FROM => image de référence
WORKDIR => change de répertoire courant
USER => change l'utilisteur courant
RUN => commandes à exécuter
ADD => copie fichier vers le container
VOLUME => Expose un répertoire pour un partage par montage
ENV => Expose une variable d'environnement
EXPOSE => Expose un port
CMD => commande ou script lancé au démarrage
ENTRYPOINT => pour utiliser docker comme une ligne de cde
```

Docker : installation d'un serveur web

- On va prendre une distribution classique

```
docker search debian:latest
```

- Puis on `pull`
- On la lance pour la tester

```
docker run --rm --name mon-debian -ti debian
```

Il faudra mettre à jour les dépots (`apt-get update`) puis installer `apache2` ainsi que `apache2-utils`

Docker : installation d'un serveur web

Créer un fichier nommé `Dockerfile` contenant les instructions suivantes

```
FROM debian:latest
RUN apt-get update
RUN apt-get install -y apache2
RUN apt-get install -y apache2-utils
RUN apt-get clean
EXPOSE 80
CMD ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
RUN mv /var/www/html/index.html /var/www/html/index-org.html
RUN echo '<h1>mon serveur apache dans docker!</h1>' > /var/ww
```

On construit le container et on le lance avec

```
docker build --rm . -t monserveur
docker run --rm --name myapache -d -p 8080:80 monserveur
# on vérifie sur 127.0.0.1:8080
```

Docker : installation d'un serveur ssh

Créer un fichier nommé `Dockerfile` contenant les instructions suivantes (attention, chaque Dockerfile doit être dans un dossier dédié !)

```
# Use a base image with Ubuntu
FROM ubuntu:latest
# Set environment variables to avoid interactive prompts durin
ENV DEBIAN_FRONTEND=noninteractive
# Update package lists and install required packages
RUN apt-get update && \
    apt-get install -y \
    net-tools \
    openssh-server \
    openssl \
    && rm -rf /var/lib/apt/lists/*
# Create the directory for SSH and set up sshd
RUN mkdir /var/run/sshd
# Configure SSH to allow root login (optional and shou
```

Une exemple complet

https://github.com/viardots/securityChallenge

Plusieurs containers

Description en yaml dans un fichier docker-compose.yml lancer `docker-compose up` Exemple :https://hub.docker.com/_/wordpress/

```
services:
  web:
    build: http
    ports:
     - "8080:80"
    links:
     - bd
  bd:
    build: bd
```

Outline

Vocabulaire

Vulnérabilité

Faiblesse dans le code

Charge utile (payload), l'action malicieuse

Détruire des fichiers, faire un déni de service, augmenter ses privilèges, obtenir un terminal (shell), installer un cheval de troie (trojan)

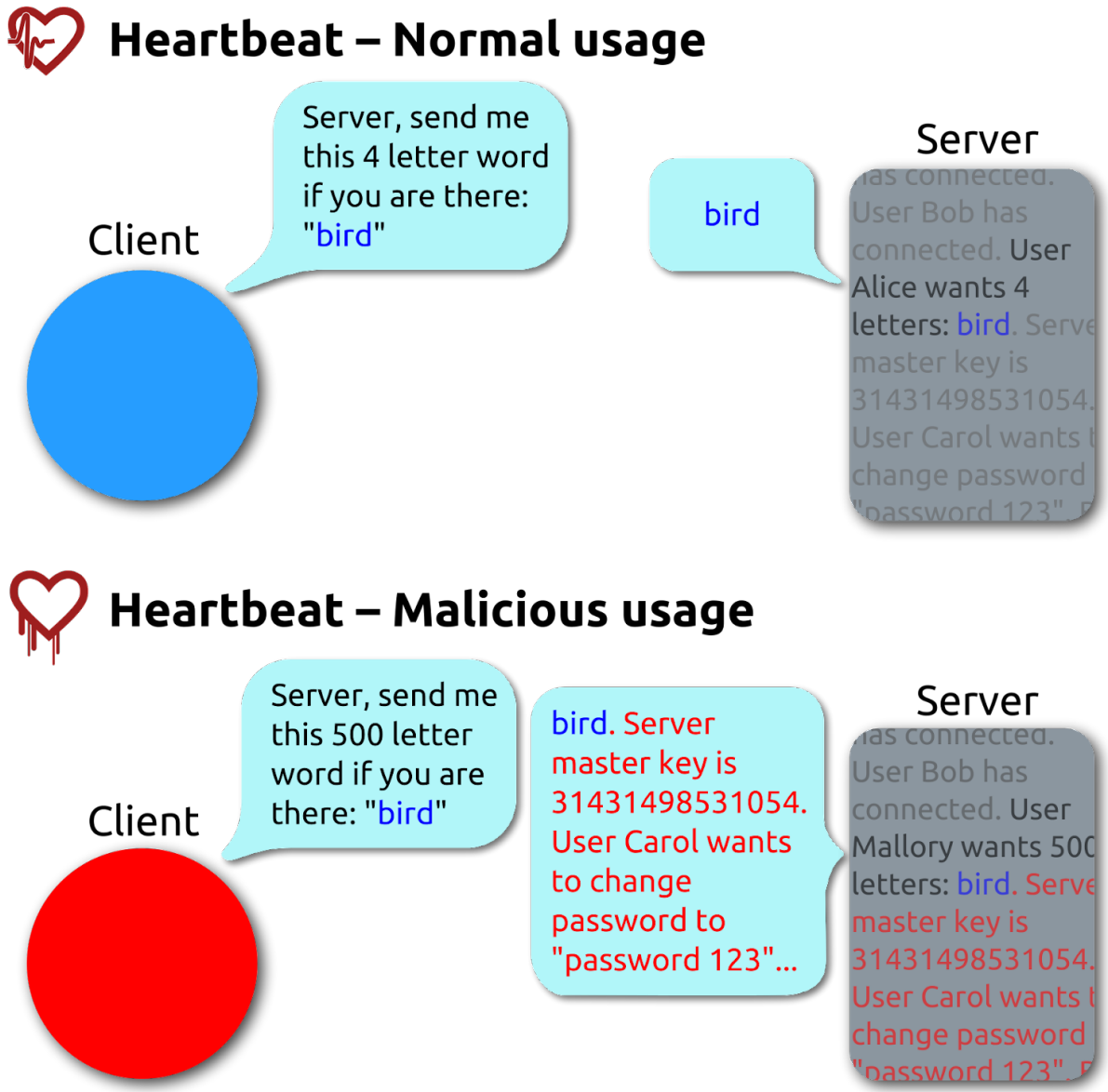
Exploitation (exploit)

Exploite la vulnérabilité pour exécuter la charge utile

Un exemple de faille : Heartbleed

Heartbleed (mars-avril 2014) a permis de récupérer 64KB de la mémoire distante du processus du serveur. Cette mémoire peut contenir la clé privé du serveur, les clés des sessions SSL/TLS ou encore les requêtes/réponses des requêtes HTTPS avec éventuellement les identifiants de connexion à l'application sous-jacentes ou les cookies de sessions.

Un exemple de faille : Heartbleed



Un exemple de faille : Heartbleed

- Concrètement, un serveur Https sur deux de la planète était concerné avec une compromission possible des clés privées, des mots de passe, de toute autre information présente en mémoire du process... Cibles vulnérables : android 4.1.1, routeurs Cisco et Juniper
- L'intégration de ce service de sécurité cryptographique a induit une vulnérabilité dont l'impact va très au-delà du périmètre de ce service
- C'est un simple oubli de vérification de bornes dans le code d'une fonction non critique du protocole SSL/TLS (introduite par un développeur bénévole et validée et intégrée par l'équipe d'OpenSSL en 2011)

faille Heartbleed : c'était pourtant si simple !

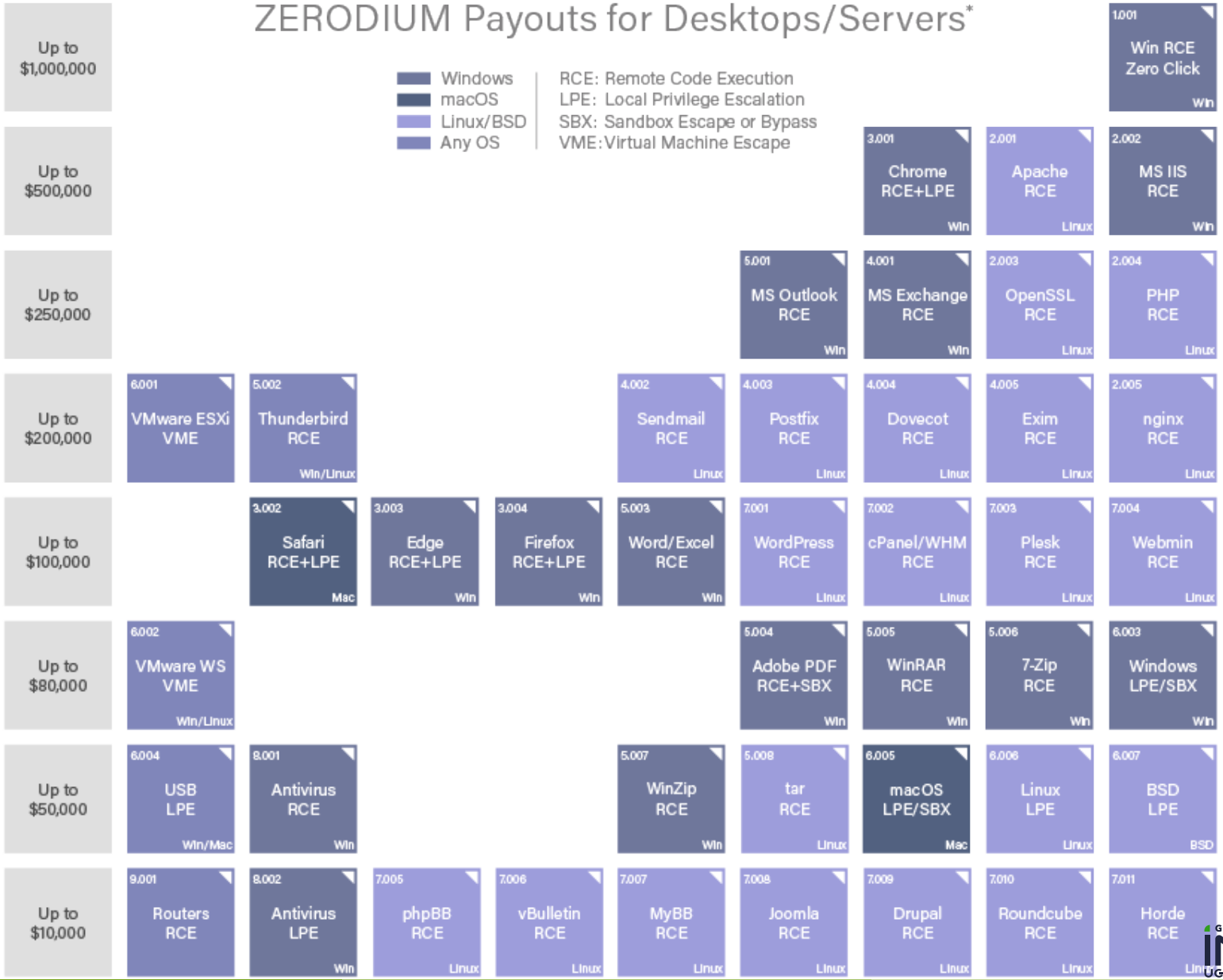
La vulnérabilité:

```
hbtype = *p++;
// macro qui prend 2 bytes de p (length(payload))
// et met dans payload
n2s(p, payload);
// length(payload) n'est jamais vérifié
pl = p;
```

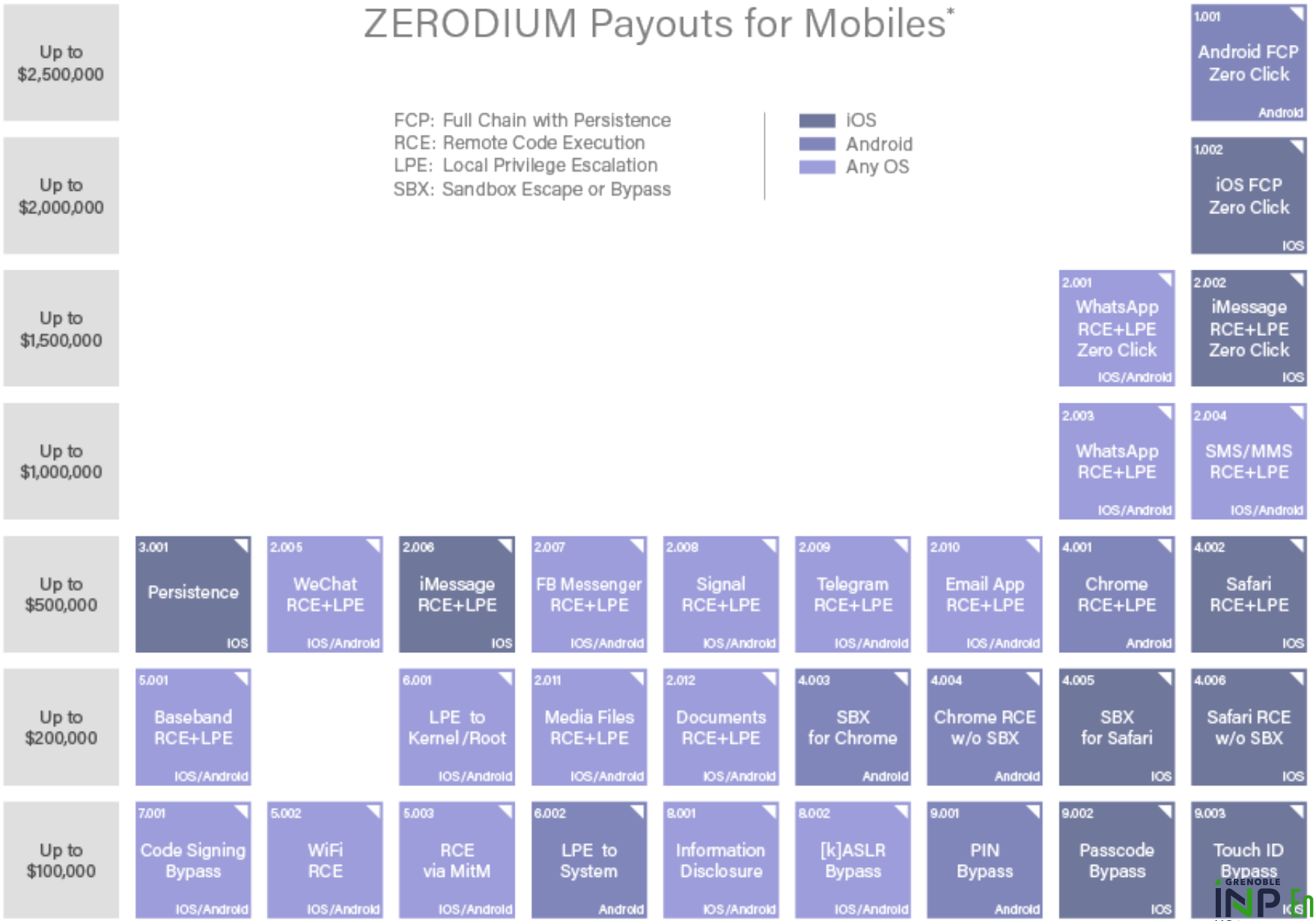
Le patch :

```
hbtype = *p++;
n2s(p, payload);
// une verification de la taille
if (1 + 2 + payload + 16 > s->s3->rrec.length) return 0
pl = p;
```

0-day vulnerabilities and récompenses 1



0-day vulnerabilities and récompenses 2



Zero Day initiative

La Zero Day Initiative (ZDI, https://www.zerodayinitiative.com) encourage à reporter de manière privée des vulnérabilités au fabricant/vendeur concerné contre récompense financière.

Outline

Recherche CVE

Il est conseillé d'utiliser les sites pour la veille de sécurité :

- https://nvd.nist.gov/vuln/search
- https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=wordpress (et remplacer par le mot clé désiré),
- https://www.cvedetails.com
- Bonnes pratiques sur le site de SSI.GOUV.FR
- Recommandation sur le développement

Exemple pratique

Un outil utile : metasploit

```
docker search msf # ou https://hub.docker.com/
docker run --rm -t -i strm/metasploit
# ou docker run -i -t awhitehatter/msf-nightly
```

Attention -> run crée un nouveau container (peut être lourd).
cf. https://github.com/awhitehatter/Dockerfiles/tree/master/msf-nightly

Exemple pratique suite

On va s'intéresser à une vulnérabilité d'une solution logicielle réelle **PHPMoAdmin**. Il s'agit d'une application web php permettant de faciliter l'administation d'une base de données MongoDB.

Lors de la parution de la faille, quelques images docker ont été créés pour pouvoir expérimenter la faille. https://github.com/ptantiku/cve-2015-2208

Exemple pratique suite

Machine vulnérable cf. cve-2015-2208

```
docker run -d -p 8888:80 ptantiku/cve-2015-2208
```

serveur http://127.0.0.1:8888/moadmin.php application accessible avec login : scott mot de passe : tiger

Exemple pratique suite

On va à présent s'intéresser à l'utilisation de Metasploit pour aller plus loin dans l'exploitation (cf. Aide mémoire Metasploit)

- Dans le container de metasploit avec la commande `search` de metasploit (cf. `help search`) identifier l'exploit réalisé dans metasploit pour cette CVE.
- Avec les commandes `use`, `info`, `show options`, `set`, `exploit`, (cf. `help CMD`), réalisons un exploit permettant de disposer à présent de l'outil meterpreter permettant de piloter la machine victime.

Exemple pratique suite

Sur metasploit :

```
search cve-2015-2208
use exploit/multi/http/phpmoadmin_exec
set RHOSTS X.X.X.X # votre adresse ip locale
# RHOST X.X.X.X si awhitehatter/msf-nightly
set RPORT 8888
set TARGETURI /moadmin.php
check
exploit
```