Sécurité numérique

Eugen Dedu

Maître de conférences
Univ. de Franche-Comté, IUT de Belfort-Montbéliard
Lpro Mobilité numérique
Belfort, France
janvier 2021

http://eugen.dedu.free.fr
eugen.dedu@univ-fcomte.fr

Objectifs

- Votre rôle professionnel : vous allez administrer des ordinateurs/réseaux, faire des applications => l'objectif du cours est de vous montrer des erreurs courantes de sécurité et comment se prémunir contre elles

- Connaissances fortement souhaitées : C, scripts shell, PHP, JavaScript, MariaDB/MySQL
- Durée : 7h CM, 13h TP
  - TP : 4h applications, 4h hachage, 4h réseau etc.
- Examen : 1h30–2h, lors de la dernière séance :
  - partie théorique sans aucune documentation
  - plus partie pratique, avec Internet à disposition

Bibliographie

- Pour des détails sur une faille, chercher sa page sur wikipedia
- Pour info, un livre sur la sécurité qui ressemble à ce cours : *Hacking: The Art of Exploitation*, par Jon "Smibbs" Erickson, avec programming, networking et cryptology comme sections, entre autres

Mise en garde

- Suivant le principe que celui qui connaît des failles ne les fait pas lui-même, ces cours et travaux pratiques vous montrent des failles de sécurité et comment se protéger contre elles
- Je décline toute responsabilité quant à la mauvaise utilisation de ces informations
- "Le fait d'accéder ou de se maintenir, frauduleusement, dans tout ou partie d'un système de traitement automatisé de données est puni d'un an d'emprisonnement et de 15000 € d'amende", et beaucoup plus dans d'autres cas [securiteinfo]

Plan

- Sécurité dans les ordinateurs et réseaux d'ordinateurs, suivant l'origine du problème :
  - à l'utilisateur, mots de passe
  - matériel
  - machine de l'utilisateur, logiciels
  - réseau, https, cryptographie
  - serveur, applications Web
- Sécurité dans l'IoT
- Domaines liés à la sécurité

Objet du cours

- Pour sécuriser les passagers d'une voiture on doit utiliser une ceinture de sécurité
- Pour sécuriser une porte on utilise une serrure
- Pour sécuriser une maison on peut utiliser des caméras
- Toute université a un fonctionnaire sécurité défense (FSD), qui gère la sécurité des agents en mission (pays à risque, rapatriement le cas échéant)
- Orange fait des formations sécurité (des chantiers...)
- Ce cours traite de la sécurité numérique : sécurité des données dans les ordinateurs et les réseaux (appareils et communication/transit)

Importance de la sécurité numérique

- Nous vivons et dépendons de plus en plus du monde numérique
- Les voleurs numériques ne touchent pas à notre voiture ou maison, mais peuvent par ex. :
  - faire des achats avec notre argent à notre insu => certains utilisent un ordinateur dédié pour se connecter à leur banque
  - crypter notre ordinateur avec toutes nos photos et méls et demander une rançon => certains sauvegardent régulièrement leurs données
  - lire nos méls ou écouter notre téléphone pour entrer dans notre vie privée ou apprendre le plan futur de notre entreprise (téléphone d'Angela Merkel écouté par USA (NSA), machine Enigma des allemands, Apple vs justice USA)
  - entrer dans nos données/ordinateur pour contacter nos amis à notre place ou lancer une attaque vers un serveur

Réalité des erreurs (bugs)

- Mais, hélas !, la programmation est trop complexe pour les gens => le code a des erreurs
- Dans le livre *Code Complete...* de Steve McConnell [mayerdan] :
  - Industry Average: "about 15-50 errors per 1000 lines of delivered code."
  - Microsoft Applications: "about 10-20 defects per 1000 lines of code during in-house testing, and 0.5 defects in released product (Moore 1992)." He attributes this to a combination of code-reading techniques and independent testing.
  - "Harlan Mills pioneered 'cleanroom development', a technique that has been able to achieve rates as low as 3 defects per 1000 lines of code during in-house testing and 0.1 defect per 1000 lines of code in released product (Cobb and Mills 1990). A few projects - for example, the space-shuttle software - have achieved a level of 0 defects in 500,000 lines of code using a system of format development methods, peer reviews, and statistical testing."
- Les bugs causent des pertes aux USA de 59 mild. $/an, correspondant à 0.6% de son PIB [wikipedia]

Vulnérabilités

- Vulnérabilité ou faille de sécurité = "A flaw or weakness in a system's design, implementation, or operation and management that could be exploited to violate the system's security policy" (IETF RFC 2828) => implications sur la sécurité, par ex. permet d'obtenir un accès non autorisé ou des privilèges ou un déni de service d'un système
- Les vulnérabilités englobent les bugs de sécurité (dans le code), mais aussi des erreurs dans la spécification (on pense de manière incorrecte que l'idée à implémenter est bonne), dans le personnel etc.
- Exemples :
  - bug normal : le PDF généré ne contient pas les lignes d'épaisseur 1 px
  - vulnérabilité : lors de la lecture d'un fichier erroné (avec des erreurs XML par ex.), le logiciel exécute le code qui se trouve dans le fichier
  - vulnérabilité : on utilise un algorithme de chiffrement des mots de passe fait-maison, inconnu des gens, mais qu'en réalité un expert peut casser rapidement
  - vulnérabilité : cas réel dans le logiciel beep
- Même si ce cours donne comme exemples des logiciels libres, tous les logiciels sont vulnérables

Exemple de vulnérabilités de debian publiées en deux semaines

Recent Debian security alerts

(4722 alerts total)

ID	Package	Date
DLA-1229-1	imagemagick	2018-01-04
DLA-1228-1	poppler	2018-01-03
DLA-1227-1	imagemagick	2018-01-01
DLA-1226-1	wireshark	2017-12-31
DSA-4076-1	asterisk	2017-12-30
DLA-1225-1	asterisk	2017-12-30
DSA-4077-1	gimp	2017-12-30
DSA-4075-1	thunderbird	2017-12-29
DLA-1224-1	mercurial	2017-12-28
DSA-4074-1	imagemagick	2017-12-28
DLA-1223-1	thunderbird	2017-12-27
DLA-1222-1	ruby1.8	2017-12-25
DLA-1221-1	ruby1.9.1	2017-12-25
DLA-1217-1	irssi	2017-12-23
DSA-4073-1	kernel	2017-12-23
DLA-1218-1	rsync	2017-12-23
DLA-1220-1	gimp	2017-12-23
DLA-1219-1	enigmail	2017-12-23
DSA-4070-1	enigmail	2017-12-21
DSA-4072-1	bouncycastle	2017-12-21

Next 20 alerts

Exemple de vulnérabilités corrigées

- Changelog de debian thunderbird 1:52.5.2-1 (24/12/2017) :
  - New upstream version 52.5.2 (22/12/2017)
  - Fixed CVE issues in upstream version 52.5 (MFSA 2017-30)
  - CVE-2017-7829: Mailsploit part 1: From address with encoded null character is cut off in message header display
  - CVE-2017-7846: JavaScript Execution via RSS in mailbox:// origin
  - CVE-2017-7847: Local path string can be leaked from RSS feed
  - CVE-2017-7848: RSS Feed vulnerable to new line Injection

Base de données de vulnérabilités

- CVE (Common Vulnerabilities and Exposures)
  - common = publique (connu publiquement)
  - exposure = problème de configuration
- Un numéro unique CVE est affecté à chaque vulnérabilité rendue publique
- Avant de rendre publique une vulnérabilité, celui qui a trouvé la vulnérabilité est prié d'informer les développeurs des produits concernés et leur laisser le temps de la corriger
- Vulnérabilité 0-day = qui n'a pas encore été rendue publique, donc inconnue
  - une vulnérabilité 0-day se vendrait entre 5 et 250 k$ en 2012 [wikipedia]

Origine des failles de sécurité

- À l'utilisateur : mdp
- Dans le matériel
- Dans le logiciel de l'utilisateur sur sa machine
  - kernel
  - bibliothèques
  - applications
  - exemples de code : buffer overflow, race condition etc.
  - support/aide du réseau : firewall, NAT
- Dans le réseau, pour les communications
  - dans la transmission des pages Web (https), ssh
  - se prémunir : cryptographie
- Dans le serveur, pour les applications distantes et les données stockées à distance
  - stockage des données (par ex. CB chez les marchands) : voir cryptographie
  - exemples : JavaScript, PHP, MariaDB/MySQL

L'utilisateur – authentification

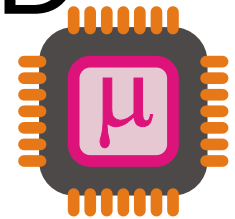
- Classiquement par mot de passe (mdp) (tapé au clavier ou graphiquement)
- 8 caractères, 64 caractères possibles => 64^8 = 10^14 possibilités
  - attaque par force brute (*brute force attack*) : il y a env. 10^5 secondes/jour, si 10^3 tests/seconde => max 10^6 jours = 3000 ans pour craquer le mdp
- Danger : mdp trop simple (du style bonjour ou année de naissance ou nom de quelqu'un)
  - attaque du dictionnaire : l'attaquant ne teste pas toutes les possibilités, mais seuls les mots du dictionnaire (100k au lieu de 10^14)
  - même si gmail par ex. vérifie, voir timisoara vs montbéliard

- John the ripper – logiciel de test et de crack de mots de passe : détecte le chiffrement utilisé et utilise l'attaque du dictionnaire, en changeant aussi légèrement les mots, ou le force brute
- Meilleure méthode de choix de mdp : prendre la première lettre de chaque mot d'une phrase (poésie par ex.)
- Ne pas être vu quand on le tape
- 1/10 personnes mettent leur mdp dans leur testament pour que leurs données puissent être utilisées [wikipedia]

L'utilisateur – renforcement de la sécurité

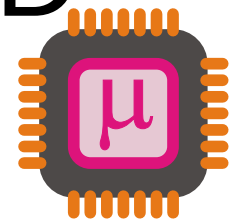
- Two-factor authentication:
  - something you know (pwd)
  - smth you have (phone, e-mail, feuille OTP one-time password)
- Third factor starts to be used:
  - smth you are (some form of biometric authentication): iris, fingerprint, pulse-response on the other hand (each human exhibits a unique response to a signal pulse applied at the palm of the other hand)
  - liveness (on ne triche pas avec une photo) – l'iris n'est pas bon pour cela, car il est statique
  - continuity (après login, c'est toujours lui qui utilise le clavier) – le truc avec hand est bien pour cela

Intel Management Engine, AMD Platform Security Processor

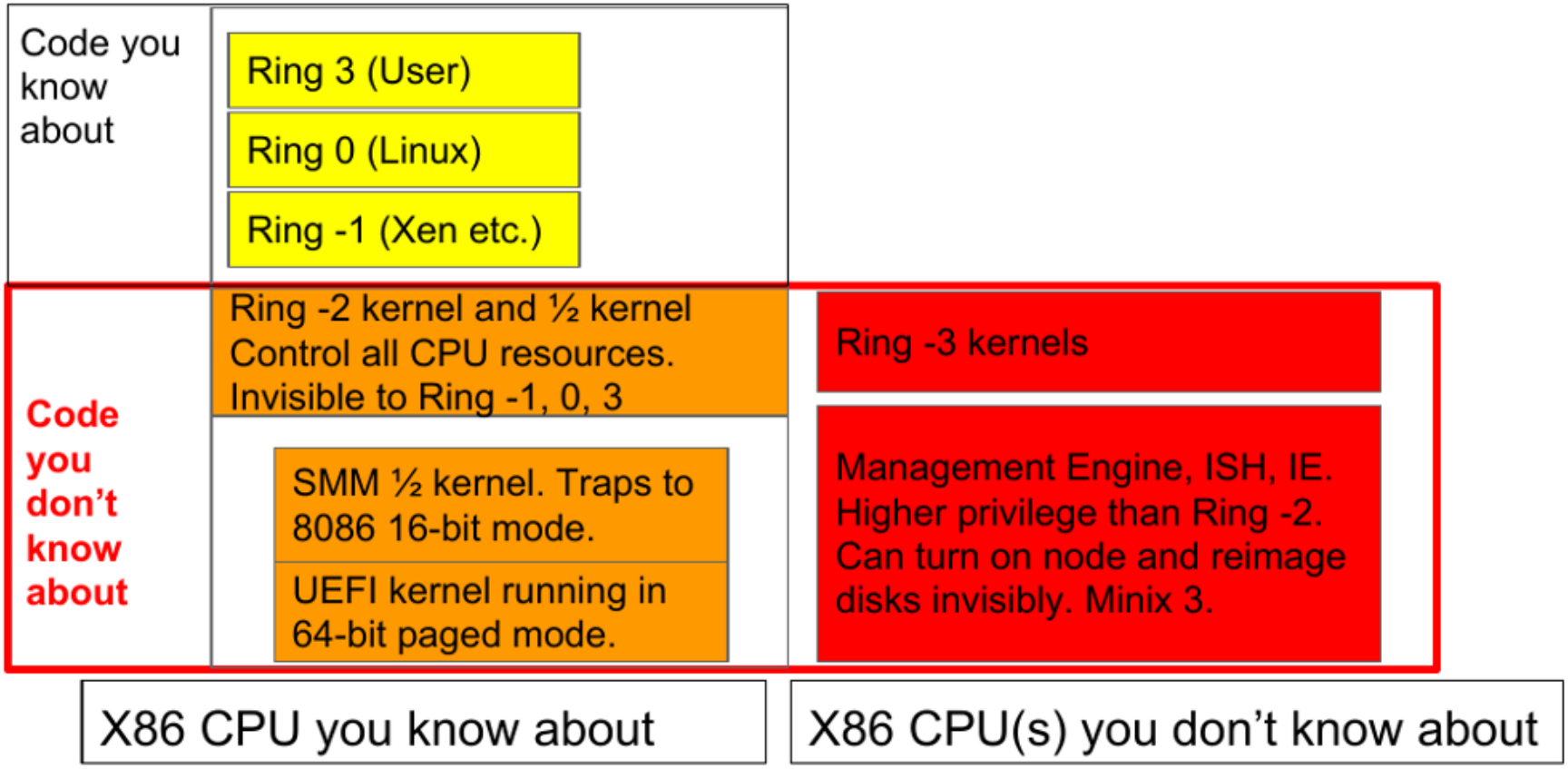


- "The Intel Management Engine (ME), also known as the Manageability Engine, is an autonomous subsystem that has been incorporated in virtually all of Intel's processor chipsets since 2008.The subsystem primarily consists of a proprietary firmware running on a separate microprocessor that performs tasks during boot-up, while the computer is running, and while it is asleep. It continues to run when the system is turned off. [...] Its exact workings are largely undocumented and its code is obfuscated using confidential huffman tables stored directly in hardware, so the firmware does not contain the information necessary to decode its contents. Intel's main competitor AMD has incorporated the equivalent technology Platform Security Processor (PSP) in virtually all of its post-2013 CPUs."
- "Zammit stresses that the ME has full access to memory (without the parent CPU having any knowledge); has full access to the TCP/IP stack and can send and receive network packets independent of the operating system, thus bypassing its firewall."
- "Several weaknesses have been found in the ME."
- Plus d'informations : [wikipedia], [theregister]

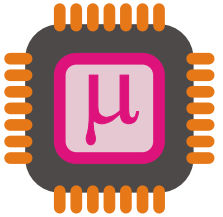
Intel Management Engine, AMD Platform Security Processor



The operating systems

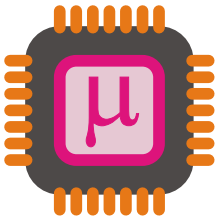


01/2018, Exécution spéculative et branch prediction (Meltdown, Spectre)



- [Vidéo démo]
- Lors d'un branch, les µP exécutent du code avant de savoir si le code est à exécuter (out-of-order execution)
- Si le branche n'était pas à exécuter, le µP revient en arrière
- Sauf que s'il a chargé dans le cache une donnée, il ne l'efface pas du cache
- => si le chargement dans le cache dépend d'une donnée, on peut l'apprendre en fonction si le cache avait été rempli ou non
- Affecte Intel, partiellement AMD, ARM, au moins
- Conséquences :
  - dans le cloud : une machine virtuelle lit dans la mémoire du hyperviseur, donc de la machine hôte
  - le Web : [mozilla] et "firefox (57.0.4-1): Fixes for mfsa2018-01, mitigating "Spectre" side-channel attack"
- "Branch prediction, speculative execution, caches are all in classic computer architecture textbooks. Whoever came up with these exploits are brilliant. I think people will find more vulnerabilities like these." – j'avais appris cela en 1998 et c'est que maintenant que cela est apparu...
- Plus d'informations : [lwn]

Vulnérabilités dans le matériel – pourquoi



- "Today's digital chips, like CPUs, usually start out as computer code written in VHDL or Verilog. They are among the most complex and intricate devices ever created." – tout comme le logiciel, le problème viendrait du fait que c'est trop complexe

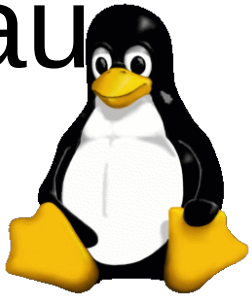
Vulnérabilités **voulues** dans le matériel

- "Nested on the servers' motherboards, the testers found a tiny microchip, not much bigger than a grain of rice, that wasn't part of the boards' original design. [...] investigators determined that the chips allowed the attackers to create a stealth doorway into any network that included the altered machines. Multiple people familiar with the matter say investigators found that the chips had been inserted at factories run by manufacturing subcontractors in China." (10/2018)
- "Still, the issue here isn't that we can't trust technology products made in China. Indeed there are numerous examples of other countries — including the United States and its allies — slipping their own "backdoors" into hardware and software products."

Vulnérabilités des logiciels, noyau

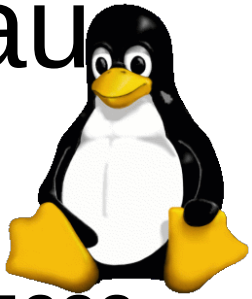
- "The Linux kernel is the source of far more CVE numbers than any other component in the system"
- "There are, Kees said, one billion Android devices in circulation [10/2015]. Most of them are running 3.4 kernels [released 05/2012]."
- Number of CVEs de linux in Ubuntu : "96 en 2012, 190 en 2013, 154 en 2014"
- Conclusion : beaucoup d'appareils Android ont des vulnérabilités connues !
  - "One Billion Android Devices Vulnerable to Privilege Escalation" (03/2014) [hackerschronicle]

Vulnérabilités des logiciels, noyau



- "A vulnerability classified as critical was found in Google Android (the affected version is unknown). Affected by this vulnerability is an unknown function of the component Kernel File System. The manipulation with an unknown input leads to a privilege escalation vulnerability. The CWE definition for the vulnerability is CWE-269. As an impact it is known to affect confidentiality, integrity, and availability."
- "The technical details are unknown and an exploit is not publicly available. The pricing for an exploit might be around USD $5k-$25k at the moment (estimation calculated on 12/06/2017). It expected to see the exploit prices for this product increasing in the near future."

Vulnérabilités des logiciels, noyau



- Linux : "Initialization-related race conditions are fairly common; another example can be seen in CVE-2015-5283. In a modular system, the module for the SCTP network protocol will not be loaded until a user requests an SCTP socket. The initialization code in the SCTP module registers its installed protocols before it is fully initialized; that opens a window within which another process can attempt to open sockets while the module is in a half-baked state. Good things rarely come from such situations."
- Présentation d'une vulnérabilité en détail (09/2018) : A cache invalidation bug in Linux memory management

Vulnérabilités des logiciels, bibliothèques

- Une bibliothèque vulnérable rend les applications qui l'utilisent vulnérables

- OpenSSL, vulnérabilité Heartbleed à partir de 03/2012, rendue publique en 04/2014
- Bug : service d'echo, la réponse ne vérifie pas la taille du message initial et utilise ce qui se trouve dans la mémoire après le message initial
- Permet de lire, par buffer overflow, dans la mémoire de la machine pour récupérer des clés privées par ex.
- 17% des serveurs Web sécurisés seraient touchés lors de la découverte
- Correction :

  ```
  if (1 + 2 + payload + 16 > s->s3->rrec.length) return 0; /* silently discard per RFC 6520 sec. 4 */
  ```

- Conséquence : d'innombrables certificats et mots de passe ont été renouvelés

Vulnérabilités des logiciels, bibliothèques

- zlib, "thousands of applications relying on it for compression, either directly or indirectly" :
  - linux (decompress image at boot time, compress file systems, compress networks protocols)
  - apache (compresse les données)
  - libpng, openssh, openssl, ffmpeg
  - subversion (compresse le traffic), git (stocke le repository)
- Out-of-bounds pointer arithmetic (CVE-2016-9840) : "inftrees.c was subtracting an offset from a pointer to an array, in order to provide a pointer that allowed indexing starting at the offset. This is not compliant with the C standard, for which the behavior of a pointer decremented before its allocated memory is undefined."
- "Compiler optimizations exist that assume code does not do this"
- Correction dans zlib : [github]

Vulnérabilités des logiciels, applications

- wpa_supplicant, l'application qui gère les connexions sans fil, utilisée par défaut sous linux (et android ?)
- "Crafted P2P SSID names could potentially be used to crash or execute code on targets"
- Comme Heartbleed, elle ne vérifie pas la longueur des données transmises
- Elle stocke le SSID dans un tableau de 32 octets, alors que la longueur maximale du SSID (stockée dans 8 bits dans le paquet) est de 255 octets
- Conséquence : un attaquant peut copier jusqu'à 223 octets de données qui débordent dans d'autres variables, et les faire exécuter
- À noter que cette faille apparaît sans action de la part de l'utilisateur (??)
- Plus d'informations [arstechnica]

Vulnérabilités des logiciels, applications

- Ekiga, logiciel libre de vidéoconférence
- 10/2012 : "ekiga before 4.0.0 allows remote attackers to cause a denial of service (crash) [...] with a party name that contains invalid UTF-8 strings"
- Le champ User-Agent était encodé en latin-1 par NetMeeting (un autre logiciel compatible), alors que GTK affiche de l'UTF-8
- Problème : non validation d'entrée
- Correction : [gnome] moi-même ☺
- Autre exemple : XML non valide

Vulnérabilités des logiciels, types de bugs

- Buffer overflow, exemple dans la suite
- Race condition, exemple dans la suite
- Erreur off-by-one
- Vérification de l'entrée (XML valide par ex.)
- Use after free (imaginons que dans la mémoire pointée par le pointeur a été mis une adresse de fonction utile à l'attaquant)
  - "For example, CVE-2014-1776 is a use-after-free vulnerability in Microsoft Internet Explorer 6 through 11 being used by zero-day attacks by an advanced persistent threat"
- Beaucoup d'autres

Buffer overflow

```
#include <stdio.h>
#include <string.h>

int main (void) {
  int pass = 0;
  char buff[8];

  printf ("Enter the password: ");
  gets (buff);

  if (strcmp (buff, "pass"))
    printf ("Wrong password\n");
  else {
    printf ("Correct password\n");
    pass = 1;
  }

  if (pass)
    printf ("Access granted\n");

  return 0;
}
```

```
$ gcc test.c
[...]
$ ./a.out
Enter the password: aaaa
Wrong password
$ ./a.out
Enter the password: aaaaaaaaa
Wrong password
Access granted
```

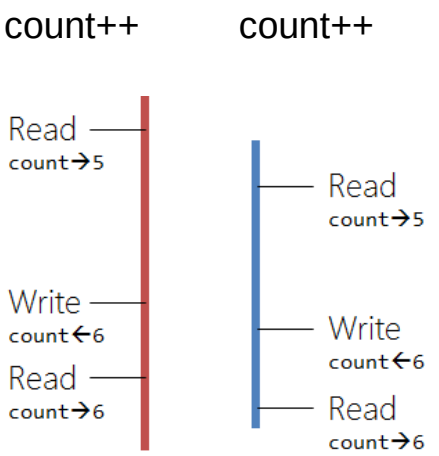
Explication : débordement de buffer dans la pile

```
ret@
pass
pass
pass
pass
buff[7]
...
buff[0]
```

Solution : remplacer gets par fgets, qui limite le nombre de caractères stockés

Notes : accents dans le mdp + gcc a des options pour empêcher cela (-fstack-protector)

Race condition



- Quand deux codes accèdent à une ressource commune en écriture et donnent des résultats différents en fonction du timing
- Très difficile de corriger, car ils se manifestent très rarement, en fonction du timing


```
function retrait ($montant) {
    $solde = getSolde ();  // de la BD
    if ($montant <= $solde) {
        $solde -= $montant;
        echo "Vous avez retiré $montant €";
        setSolde ($solde);
    } else
        echo "Pas suffisamment d'argent";
}
```

Explication :
- montant initial de 1000 € et deux appels en même temps
- ressource partagée et code qui l'utilise

Solution : utiliser un lock (mutex) ou des opérations atomiques sur la BD

Se prémunir des vulnérabilités des logiciels

- Mettre à jour régulière les ordinateurs, surtout les serveurs (qui sont publiques)
  - sous debian : apt-get update && apt-get upgrade
- Obtenir une "carte" d'un réseau (nmap) => fermer les services inutiles
- Firewall, sur le réseau ou sur la machine : spécifie les communications autorisées
- NAT, sur le réseau : empêche les connexions depuis l'extérieur

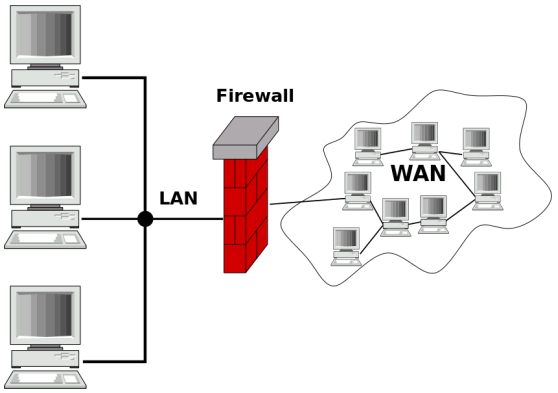
Carte de "services" d'un réseau

- nmap (network mapper) crée une "carte" du réseau :
  - découvre les machines
  - leurs services et leurs versions
  - leurs OS et matériel
- Cela permet à un attaquant de savoir où et comment attaquer
- Solution : réduire les portes d'entrée de l'extérieur en fermant les services (ports) inutiles ou les laisser ouverts que pour certaines machines (par ex. localhost)

```
snoopy:~$ nmap xyz
[...]
Not shown: 997 closed ports
PORT    STATE SERVICE
25/tcp  open  smtp
80/tcp  open  http
631/tcp open  ipp

Nmap done: 1 IP address (1 host up)
scanned in 0.15 seconds
snoopy:~$ nmap -A xyz
[...]
```

Pare-feu (firewall)



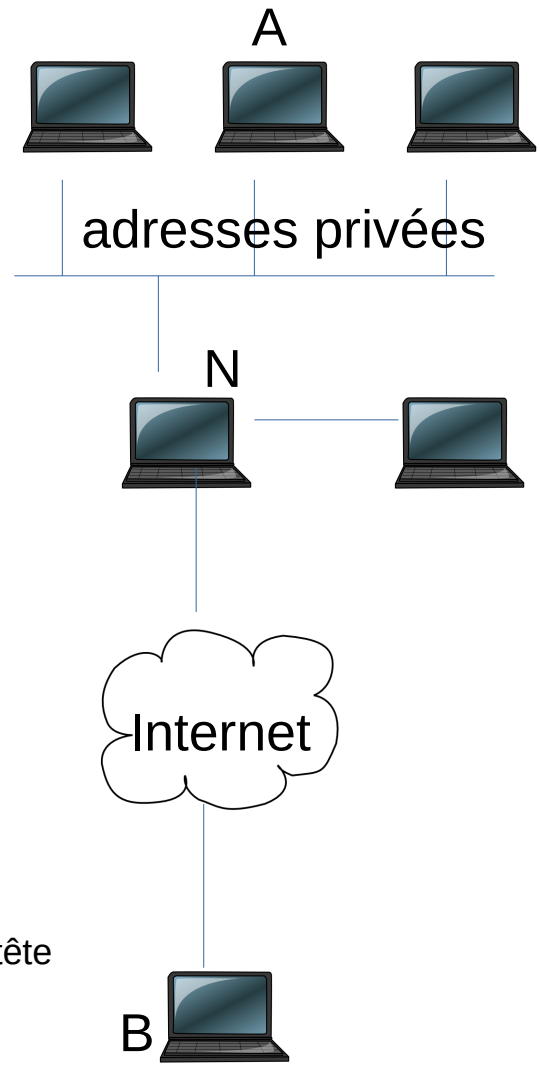
- Spécifie les types de communications autorisées
- Pare-feu réseau :
  - sans état, access control list (ACL), laisse passer que certains ports par ex., rend inutilisable un serveur (avec ses vulnérabilités) sur une machine d'utilisateur par ex.
  - avec état, laissent passer que les paquets qui s'enchaînent dans une communication
  - applicatif, vérifie la conformité du paquet au protocole (ne marche pas pour HTTPS)
  - souvent fait office de machine NAT aussi (transp. suivant)
- Pare-feu personnel (sur la machine même) :
  - vérifie aussi le logiciel qui initie une connexion

Network Address Translation (NAT)

- A (adresse privée) envoie un paquet à B
- N intercepte le paquet de A
- N crée une correspondance entre A et B
  - crée aussi un port P pour cette connexion
- N renvoie le paquet à B en lui changeant :
  - l'adresse source IP : sa propre adresse
  - le port source TCP/UDP : P
- Retour et autres paquets : utilisation de la correspondance
- Machine NAT : PC, routeur, firewall
- La machine qui fait la NAT maintient un tableau :

srcaddr	srcport	protocol	natport
10.1.2.3	9845	TCP	45320
10.1.2.3	9856	TCP	47099
10.1.2.4	7455	TCP	32455
10.1.2.4	7455	DCCP	32456

- Impossible d'initier une connexion depuis l'extérieur
- Certains protocoles haut-niveau (FTP, SIP, ...) mettent des adresses IP dans leur en-tête



Problèmes posés par le réseau pour les communications

- On souhaite se prémunir contre les attaques provenant du réseau (sur un lien ou sur un appareil réseau) pour nos communications
- On ne peut pas empêcher quelqu'un (E) de **voir** le message entre A et B si elle se trouve sur le chemin, mais on peut rendre cela inutile
- Propriétés souhaitées :
  - confidentialité : E **ne peut pas comprendre** notre message (communication sécurisée, confidentielle)
  - intégrité : si E modifie le message, B s'en rend compte
  - authenticité : A est sûr qu'il discute avec B (et non avec E)
- Solution : la cryptographie

- Intégrité (en local) : le fichier que nous avons sur un ordinateur n'a pas été modifié (hash)

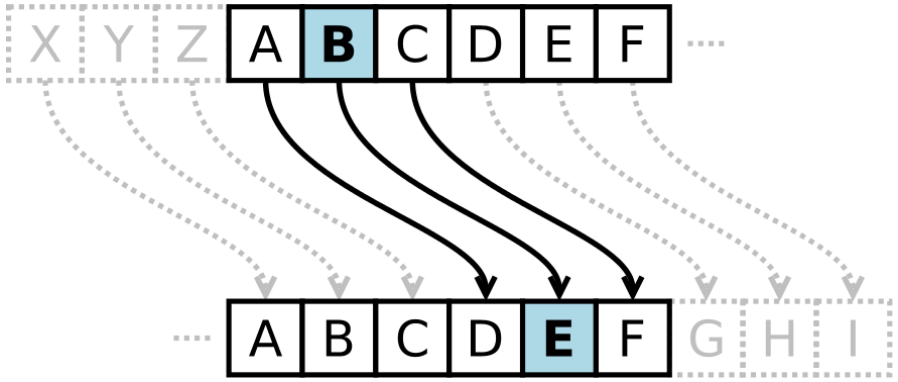
Cryptographie

- Principe de Kerckhoffs : la sécurité d'un système ne doit reposer que sur le secret de la clé ; tous les algorithmes doivent être publics
- *Security by obscurity* n'est pas efficace ; il faut *security by state of the art cryptography*

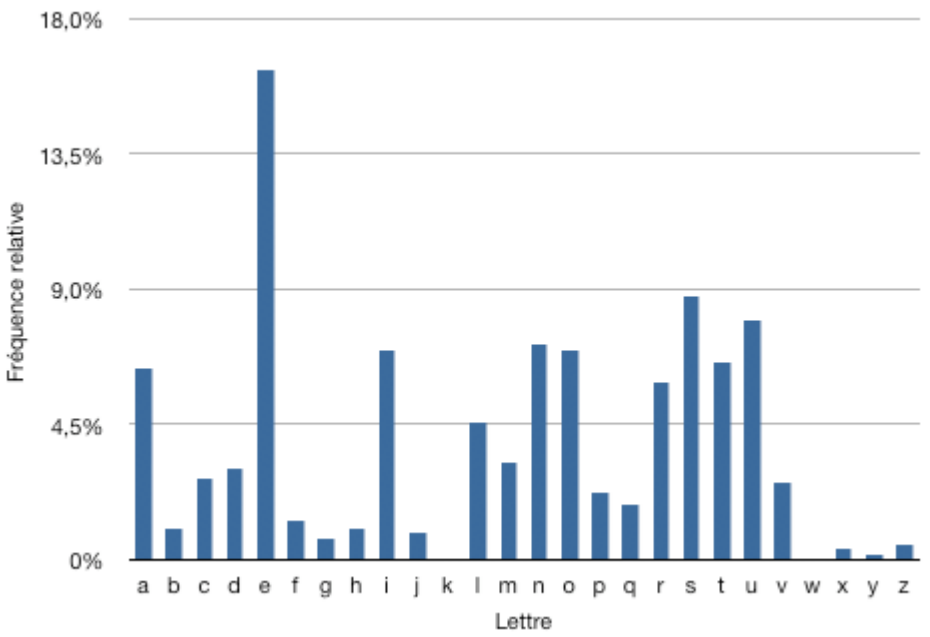
- Chiffrements symétrique et asymétrique
  - but : communication sécurisée (confidentialité), on souhaite pouvoir décrypter le message
  - autres buts : authentification, intégrité
  - TLS, HTTPS, cryptage Wi-Fi
- Fonction de hachage (hash)
  - but : intégrité, on ne souhaite pas retrouver le texte clair
  - taille fixe du message hashé

Chiffrement symétrique (à clé secrète) – code de César

```
original : WIKIPEDIA L'ENCYCLOPEDIE LIBRE
encodé   : ZLNLSHGLD O'HQFBFORSHGLH OLEUH
```



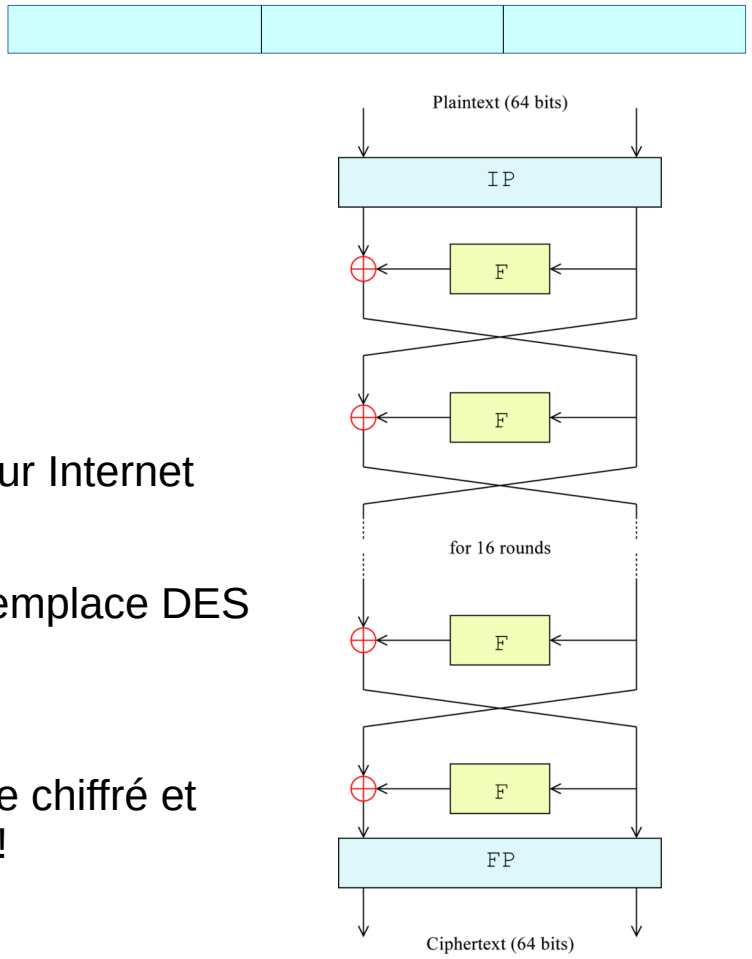
Attaque : analyse fréquentielle



Chiffrement symétrique (à clé secrète) – DES

- DES (Data Encryption Standard)
  - F est la fonction Feistel, non présentée dans ce cours
  - blocs de 64 bits, et clé de 56 bits + 8 bits de parité
- Historique :
  - 1977 DES devient standard aux USA
  - 1997 DES cassé pour la première fois
  - 1998 cassé en 56 heures avec une machine dédiée
  - 1999 cassé en 22 heures avec la machine + en distribué sur Internet
  - 1999 3DES (applique 3 fois DES avec 3 clés) est préféré
  - 2001 AES (bloc de 128 bits, clé de 128, 192 ou 256 bits) remplace DES comme standard aux USA
- Propriétés :
  - même si on connaît l'algorithme de chiffrement, le message chiffré et son message initial, on ne peut toujours pas trouver la clé !

Message à chiffrer, découpé en blocs de 64 bits

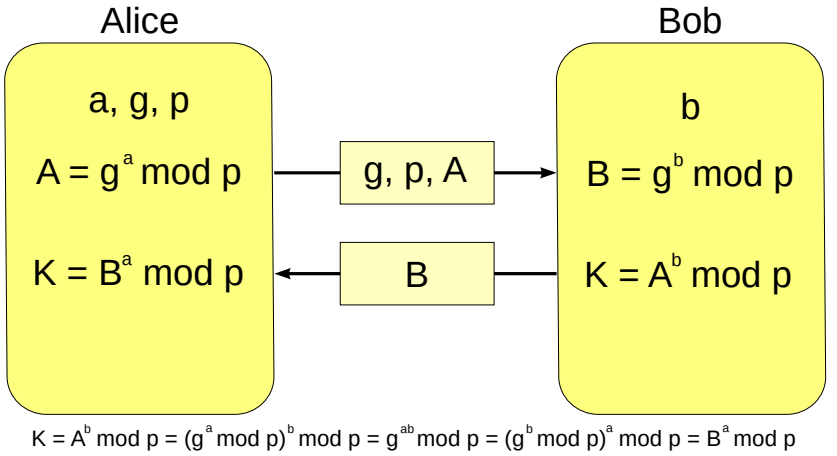


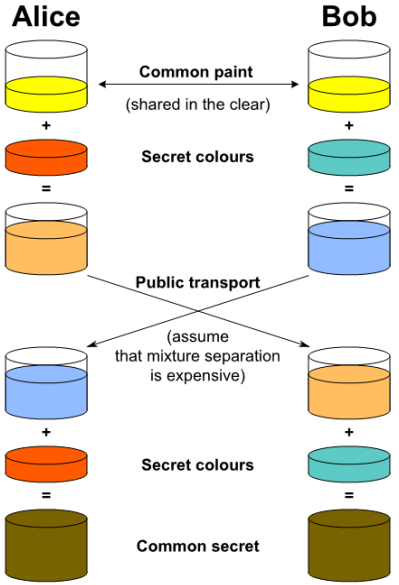
Chiffrement symétrique (à clé secrète)

- Une seule clé pour chiffrement et déchiffrement
- Les deux personnes doivent connaître la clé
- Exemples : crypter des fichiers sur le disque dur, crypter la communication entre un ordinateur et son point d'accès, faire communiquer deux personnes qui ont une clé commune
- Génération de la clé : nombres aléatoires
  - les RNG en logiciel ne sont pas 100% aléatoires (pseudo-aléatoire)
  - pour forte sécurité on utilise le bruit de l'environnement, donc matériel (timing des appuis sur touches, divers événements apparaissant dans l'ordinateur)
- Inconvénient : échange de la clé :
  - valise diplomatique
  - protocole Diffie-Hellman
- Propriétés :
  - assure confidentialité, mais pas l'intégrité, ni authentification

Génération de clé : protocole Diffie-Hellman

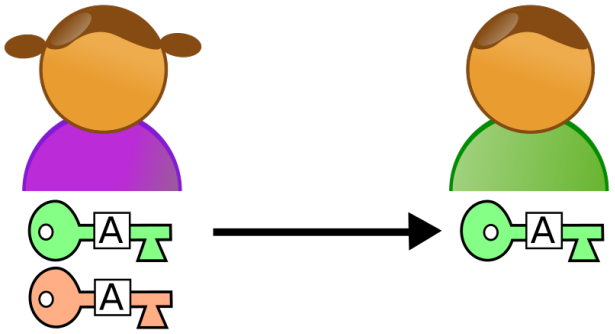
- But : se mettre d'accord sur une clé secrète en utilisant un canal non sécurisé
  - on peut ensuite utiliser la clé pour chiffrement symétrique
- Hypothèse : mixer est rapide, séparer est trop long
- N'assure pas l'authentification => vulnérable à une attaque man-in-the-middle (l'homme du milieu)
  - il intercepte tout et utilise deux "couleurs" pour les deux personnes
  - il continue à être au milieu, sinon l'attaque est découverte



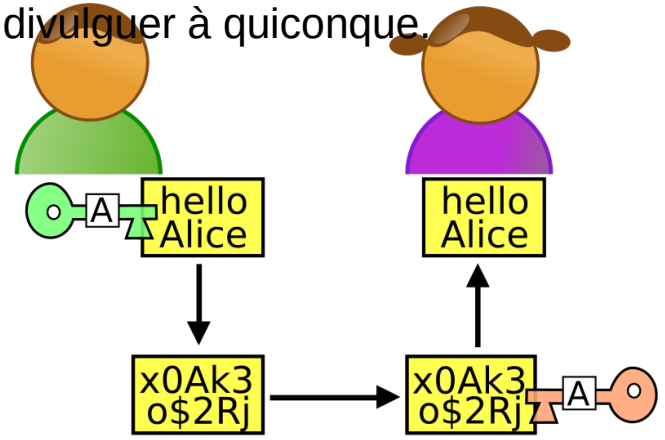


Chiffrement asymétrique (à clé publique)

- Utile quand on n'a pas de canal sûr pour s'échanger une clé secrète ou quand on communique avec beaucoup d'utilisateurs (https par ex.), car on ne peut pas avoir un mdp différent avec chacun
- Deux clés : une privée et une publique
  - la clé publique est déduite de la clé privée (??), par une fonction à sens unique
  - exemple de fonction : factorisation en nombres premiers, C=p1*p2
- A chiffre avec la clé publique de B => seul B peut décrypter
- A peut chiffrer ensuite avec sa clé privée => B saura que c'est bien A l'expéditeur
- Moins performant que le chiffrement symétrique : temps de traitement plus long, et clé plus longue pour sécurité équivalente
- Souvent utilisé juste pour s'échanger une clé (ex. : Diffie-Helmann), ensuite pour la communication on utilise le chiffrement sym avec cette clé
- RSA, clés de 1024 à 4096 bits
- Utilisation : se loguer avec ssh sans mdp, être dans le web of trust etc.



1ère étape : Alice génère deux clés. La clé publique (verte) qu'elle envoie à Bob et la clé privée (rouge) qu'elle conserve précieusement sans la divulguer à quiconque.



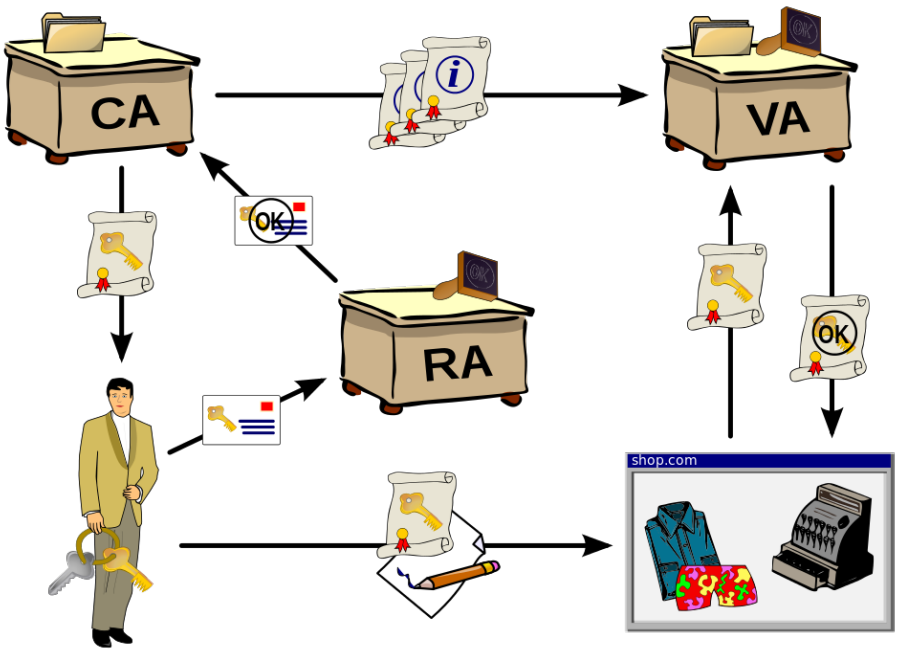
2ème et 3ème étapes : Bob chiffre le message avec la clé publique d'Alice et envoie le texte chiffré. Alice déchiffre le message grâce à sa clé privée.

Authentification – PKI

Public key infrastructure (PKI), infrastructure à clé publique

- Centralisé
- Le propriétaire d'un site S génère sa clé privée, et à partir d'elle son certificat
  - certificat = clé publique + adresse etc.
- Une autorité de certification (CA), en qui tout le monde a confiance et dont le certificat est connu de tous, stocke des certificats

- Tout navigateur a une liste de CA, voir Preferences->Privacy&Security->Certificates
- Quand il contacte un site S, il vérifie que le certificat qui lui est donné est le même que le certificat de S donné par la CA



CA = autorité de certification
VA = autorité de validation
RA = autorité d'enregistrement (vérifie que le certificat généré appartient à la personne prétendue)

Authentification – web of trust

- Décentralisé
- Key signing parties, avec vérification des papiers d'identité et échange de clés
- Chacun est responsable d'utiliser le certificat de l'autre

TLS (Transport Layer Security), HTTPS

- TLS se met entre le protocole de transport et l'application
- TLS fournit :
  - cryptage et intégrité des données : par chiffrement symétrique
  - authentification du serveur : par infrastructure à clés publiques
- Applications :
  - HTTPS = HTTP Secure = HTTP sur TLS
    - configurer le serveur Web pour utiliser HTTPS, port 443 par défaut [onlamp]
  - programmes de courriel, visioconférence (VoIP), IM etc.
- Plus d'informations : [wikipedia]

Cryptage Wi-Fi

- But : rendre la communication sur le médium sans fil aussi sécurisée que le médium filaire
- Certaines applications cryptent leur communication (https, ssh, pop3s), mais pas toutes (http, pops, smtp, ftp)
- WEP : chiff. sym. RC4 (clé de 40 ou 104 bits), intégrité par CRC-32, et 2 méthodes d'authentification
  - utilisé à partir de 1997, cassé en 2001
  - remplacé temporairement par WPA
- WPA2, utilisé depuis ~2004
- WPA3, annoncé en 01/2018, clé de 128 ou 192 bits

Hash

- Hash = fonction qui mappe une donnée de taille arbitraire en une donnée de taille fixe
- À ne pas confondre avec somme de contrôle, compression sans perte, code correcteur d'erreur etc.
- Fonction cryptographique de hachage = hash avec des propriétés de sécurité :
  - impossible de trouver le message à partir du hash (fonction à sens unique, *one-way function*)
  - impossible en pratique de trouver deux messages donnant le même hash (collision)
    - deux messages différant d'1 bit donnent des hashs très différents
  - le calcul du hash est rapide

- Exemples :
  - MD5, SHA-1 – cassés, à ne pas utiliser
  - SHA-2 (avec hashs de taille 224, 256, 384 or 512 bits), SHA-3

```
$ sha224sum vmlinuz-4.14.0-2-amd64
7d4e7041ebceeaa70b05d81b04d9bf15eb9ad7f546cbc98e3617799a  vmlinuz-4.14.0-2-amd64
$ echo 0 >>vmlinuz-4.14.0-2-amd64
$ sha224sum vmlinuz-4.14.0-2-amd64
91c22a786ad14d25e2965959bb5bff54a1e2fd628950a8fc422c0e95  vmlinuz-4.14.0-2-amd64
```

Hash, sel, poivre – utilisation

- Hash seul :
  - on stocke hash (pwd)
- Sel (*salt*) :
  - créé automatiquement lors du choix du mdp, un par mdp
  - stocké à côté du hash, publique
  - on stocke hash (pwd + sel)
  - but : augmente le nombre de possibilités, donc évite des attaques basées sur table de hachage (*rainbow table*)
  - utilisé pour le mdp dans linux depuis le début
- Poivre (*pepper*) :
  - un seul (ou quelques-uns, testés l'un après l'autre) pour tout le programme
  - secret, stocké à un autre endroit
  - on stocke hash (pwd + sel + poivre)

Différences entre cryptage et hachage

- "A hash is a string or number generated from a string of text. The resulting string or number is a fixed length, and will vary widely with small variations in input. The best hashing algorithms are designed so that it's impossible to turn a hash back into its original string."
- "Encryption turns data into a series of unreadable characters, that aren't of a fixed length. The key difference between encryption and hashing is that encrypted strings can be reversed back into their original decrypted form if you have the right key."
- "Encryption should only ever be used over hashing when it is a necessity to decrypt the resulting message. For example, if you were trying to send secure messages to someone on the other side of the world, you would need to use encryption rather than hashing, as the message is no use to the receiver if they cannot decrypt it. If the raw value doesn't need to be known for the application to work correctly, then hashing should always be used instead, as it is more secure."
- [securityinnovationeurope]

Que choisir entre cryptographie symétrique/asymétriq. et hachage ?

- A veut communiquer avec son frère
- A veut communiquer avec quelqu'un qu'il ne connaît pas
- A a écrit un logiciel et envoie sur une liste de diffusion l'annonce, il souhaite que les utilisateurs puissent vérifier que le logiciel qu'ils téléchargent est le bien le sien (n'a pas été modifié sur le serveur)
- A souhaite stocker des mdp et s'assurer que même si sa BD est cassée, les mdp ne sont pas retrouvés
- A souhaite vérifier régulièrement qu'aucun des fichiers d'un certain répertoire n'a été modifié
- A souhaite vérifier si un de ses répertoires est un miroir d'un répertoire d'une autre machine lointaine

Cas réel à discuter

- Montrer un OM et le remplissage d'une mission
- Comment donner accès à quelqu'un d'extérieur à une entrée de la BD ?

Sécurité des sites Web

- Internet est un endroit dangereux !
- Exemples :
  - British Airways lance une enquête après le piratage de 380.000 cartes de paiements en ligne (09/2018)
  - « Anyone could access the entire database, including real time child location, name, parents details etc. Not just Gator watches either – the same back end covered multiple brands and tens of thousands of watches » (GPS-enabled children's watches, 02/2019)
  - Facebook a stocké des centaines de millions de mots de passe en clair, et ses employés y avaient accès (03/2019)
- Attaques sans passer par les pages Web :
  - sécurité du serveur Web et de la BD (des données)
- Attaques via les pages Web :
  - race condition, attaque du dictionnaire peuvent apparaître aussi
    - captcha – immunise contre l'attaque du dictionnaire ; de plus on s'assure que c'est un utilisateur derrière, pas une machine ultra-rapide
  - injection SQL
  - cross-site scripting (XSS)
  - cross-site request forgery (CSRF)

Stockage des données

- Les sites Web stockent parfois des informations sensibles : cartes bancaires, mdp etc.
- Il est possible d'attaquer directement le serveur ou la BD, sans passer par le site Web
- Solutions :
  - crypter ses données (la BD)
  - le hachage permet de vérifier l'intégrité des données (si les pages Web ont été changées par ex.)

Injection SQL

- Attaque :
  - soit une page avec un champ texte appelé nom pour le nom à chercher
  - le code PHP suivant est vulnérable :

    $req = "SELECT * FROM produits WHERE nom='" . $_GET['nom'] . "'";
    - attention : mettre toujours des apostrophes aux valeurs !
  - si l'utilisateur met comme nom :

    x' OR 1='1 => il récupère toute la table !

- Solution : en PHP, au lieu d'utiliser $_GET['nom'], utiliser :

  mysqli_real_escape_string (..., $_GET['nom'])
- Autre solution : requêtes préparées

Cross-site scripting (XSS)

- L'attaquant injecte des scripts qui seront, via le site Web, exécutés dans les navigateurs d'**autres** utilisateurs
- Comme cela vient du site Web, le navigateur fait confiance au code, et lui permet par ex. d'accéder aux cookies du site ou de rediriger vers un autre site
- XSS réfléchi (non permanent) :
  - on envoie le lien ...?comment=<script src="..."></script> à l'utilisateur, par mél ou un site social
  - quand le site affiche le paramètre, le script s'exécute avec les droits de l'utilisateur
- XSS stocké (permanent) :
  - un site où les utilisateurs, après login, peuvent ajouter des commentaires (forum)
  - le méchant remarque que s'il met comme commentaire J'aime les fleurs !<script src="...">, le texte s'affiche et le script s'exécute
  - chaque utilisateur qui regarde la page verra le texte, et le script s'exécutera, à son insu
- Dans les deux cas, le script récupère le cookie d'authentification et l'envoie sur le serveur du méchant, lui permettant désormais de se faire passer pour l'utilisateur
- Cas réels : bugzilla
- Solution : vérifier les entrées des utilisateurs

Cross-site request forgery (CSRF)

- L'utilisateur exécute une action sur un site (par ex. faire un transfert d'argent) à partir d'ailleurs (un **autre** site ou un mél)
- A sait que le site S transfère de l'argent avec http://S?comptedest=X&somme=Y
- A crée un site avec un formulaire contenant des champs cachés avec comptedest et somme, ainsi qu'un bouton "Deviens riche"
- A envoie le lien vers son site à tous les utilisateurs du site S qu'il connaît
- Résultat : quand un utilisateur clique sur le bouton, le transfert se fait
- Note : A n'a pas besoin de récupérer les cookies des utilisateurs
- Solution : pour faire le transfert, le serveur peut utiliser un texte secret (*token*), généré par le site même dans la page de transfert

Sécurité en PHP

- Ne jamais faire confiance aux valeurs reçues de l'utilisateur :
  - vérifier toujours ces valeurs avant utilisation, par ex. entier entre 2 et 6
  - pour afficher une valeur : utiliser strip_tags pour enlever les balises, htmlspecialchars et htmlentities pour remplacer les caractères spéciaux
  - utiliser preg_match pour accepter que les chaînes avec certains caractères seulement
- D'où viennent les valeurs des utilisateurs ? Cela dépend de votre page :
  - paramètres de formulaires (GET ou POST)
    - attention : même si un combobox est utilisé ou qu'une vérification JS soit faite, l'utilisateur peut modifier lui-même l'URL, par ex. : ...?name=Jean
  - BD, cookies, autres pages Web etc.
- Où le problème peut-il se manifester ?
  - chez les autres utilisateurs (capture de leurs mots de passe)
  - sur le serveur (effacement BD)

Malwares : virus, ver, ransomware

- Virus : se propage via des actions de l'utilisateur, se met dans un fichier
- Ver : se auto-propage (sans action de l'utilisateur) via le réseau (ou pages Web, méls) en exploitant des vulnérabilités, habituellement ne se met pas sur le DD de la machine, pour ne pas être trouvé
- Propagation :
  - cheval de Troie : un logiciel légitime qui a été modifié pour y insérer un code malveillant, que l'utilisateur installe de son propre gré
  - téléchargement d'une version cassée d'un logiciel payant sur un site inconnu, mise-à-jour d'un logiciel, ouverture d'une pièce jointe
- Actions pour les deux :
  - lisent le clavier pour déduire le mdp gmail par ex.
  - se faire passer pour un autre pour envoyer des méls ou attaquer à plusieurs un site (déni de service)
- Ransomware : bloque l'accès à l'ordinateur et demande une rançon pour le débloquer
- Antivirus : avec signature de virus et sans (heuristiques)

Your personal files are encrypted by CTB-Locker.

Your documents, photos, databases and other important files have been encrypted with strongest encryption and unique key, generated for this computer.

Private decryption key is stored on a secret Internet server and nobody can decrypt your files until you pay and obtain the private key.

You only have 96 hours to submit the payment. If you do not send money within provided time, all your files will be permanently crypted and no one will be able to recover them.

Press 'View' to view the list of files that have been encrypted.

Press 'Next' for the next page.

WARNING! DO NOT TRY TO GET RID OF THE PROGRAM YOURSELF. ANY ACTION TAKEN WILL RESULT IN DECRYPTION KEY BEING DESTROYED. YOU WILL LOSE YOUR FILES FOREVER. ONLY WAY TO KEEP YOUR FILES IS TO FOLLOW THE INSTRUCTION.

View

95:59:59

Next >>

Sécurité dans l'IoT

- Importance : l'IoT amène beaucoup d'appareils qui ne sont pas bien protégés, on ne connaît pas ce qu'ils font et auxquels on n'a pas accès
- Ils sont beaucoup moins sécurisés qu'on ne puisse l'imaginer
- Souvent, il n'est pas de moyen pour actualiser ("upgrader") le code qui se trouve dans l'appareil => quand on bug est découvert, il est impossible de le "patcher" et reste vulnérable pour toute sa vie
- Ils se développent, et le nombre d'appareils vulnérables augmente
- Domaines :
  - systèmes financiers
  - équipement industriel
  - aviation, automobile
  - systèmes médicaux

Exemples d'appareils

- Appareils industriels : Stuxnet, le virus pour les centrifugeuses d'Iran
- Autocommutateur téléphonique privé (PABX)
  - "we keep deploying new devices that are insecure-by-default"
  - mot de passe par défaut
  - stage d'étudiant RT de ~2010 sur les failles des autocommutateurs
  - l'attaquant se connecte à l'autocommutateur et appelle une centrale téléphonique à l'étranger à 2 €/min le vendredi soir
- TV Samsung : on peut le mettre à l'écoute et envoyer l'écoute, sans que le propriétaire s'en rende compte
- Assistants vocaux, Amazon Echo, Google Home : avec un accès physique, un attaquant peut transmettre tout ce qu'il entend à son site Web de manière inaperçue
  - A Hacker Turned an Amazon Echo Into a 'Wiretap' [wired]
- "some firms are developing light bulbs that serve as Internet hubs, relying Wi-Fi signals", mais ils ne sont pas bien sécurisés
- "The Diqee 360 robotic vacuum cleaner can be turned into a surveillance device. The attack requires physical access to the device[...] But why in the world is the vacuum equipped with a microphone?" (08/2018)

Mirai botnet

- Paru sep. 2016
- Affecte home routers, digital video recorders, and webcams
- Ne peut pas être corrigé, à part remplacer l'appareil par un autre nouveau
- The Botnet That Broke the Internet Isn't Going Away [wired]
- "Much of the internet unavailable for millions of people"
- "Mirai is a type of malware that automatically finds Internet of Things devices to infect and conscripts them into a botnet—a group of computing devices that can be centrally controlled. From there this IoT army can be used to mount distributed denial of service (DDoS) attacks in which a firehose of junk traffic floods a target's servers with malicious traffic. In just the past few weeks, Mirai disrupted internet service for more than 900,000 Deutsche Telekom customers in Germany"
- Overall, the security of the average Internet-of-Things device is so bad that this attack is in the noise compared to the previously known risks [schneier]
- Imaginez ce qui peut se passer avec les bugs de sécurité des logiciels des voitures ou des implants médicaux...

Stuxnet

- Le premier ver qui endommage des systèmes industriels et le premier qui inclut un rootkit PCL
- Découvert en 2010, en action depuis 2005 paraît-il
- Infecte 200000 ordinateurs et endommage 1000 machines
- Cible : programme nucléaire iranien
- Se propage que dans certaines machines et injecte le code destructeur que dans les ordinateurs contrôlant un PLC (automate programmable industriel) avec le logiciel Siemens Step7
- Fait tourner les centrifugeuses plus vite qu'elles peuvent résister => aurait détruit 1/5 des centrifugeuses iraniennes (d'autres pensent que cela n'a pas eu trop d'effet ?!)
- Utilise 4 vulnérabilités 0-day !!
- Extrêmement sophistiqué, ce serait le ver le plus coûteux en temps de développement, et on pense que l'auteur est NSA/Israël
- Source : [wikipedia]

Moteur de recherche de failles de sécurité

- Shodan, lancé en 2009, scanne les ports http(s), mais aussi ftp(21), ssh(22), telnet(23), snmp(161), sip(5060), rtsp(554)
  - rtsp permet d'accéder à des webcams et leur flux vidéo
- A trouvé des feux de circulation, caméras de sécurité, stations de gas, usine de traitement d'eau, réseaux d'énergie, turbines de vent, centrales nucléaires, cyclotrons d'accélération de particules etc., beaucoup d'entre eux étant peu sécurisés :
  - beaucoup d'entre eux utilisent admin/1234
  - feu : on peut le mettre en mode test
- "Researcher Eireann Leveret used Shodan to identify more than 100 000 vulnerable IoT devices in 2011"
  - most of them are special-purpose industrial computers, allowing to control systems by remote supervisory staff

Importance et perception de la sécurité par les utilisateurs d'IoT

- Les clients de systèmes industriels souvent voient la couche de sécurité plus une contrainte ("burden") qu'un bénéfice, par ex. il faut se loguer pour y avoir accès
  - "Security and convenience rarely go hand-in-hand"
- La sécurité est souvent pas considéré un problème important, et ils sont plus intéressés par availability and reliability
- "Industrial control systems have relied on security by obscurity" (Mather(ly John??))
- Beaucoup d'utilisateurs préfèrent la convenance (facilité d'utilisation) et ne tiennent pas compte de la sécurité
- Étudiant 2017 : il y a déjà beaucoup de vulnérabilités ailleurs, donc une vulnérabilité de plus ne change rien...
- "Now that software is appearing in durable goods, such as cars and medical devices, that can kill us, software engineering will have to come of age"

Domaines liés à la sécurité
Stéganographie / stéganalyse

- steganós graphế = écriture protégée
- Dissimuler/cacher un message dans un autre message, mais sans le protéger
  - une application : watermarking (tatouer l'auteur)
- Analogies :
  - cryptographie = enfermer son argent dans un coffre-fort
  - stéganographie = le cacher dans le jardin
- Stéganalyse :
  - par analyse statistique : une caméra fait des photos avec certains artéfacts, qu'on peut vérifier s'ils sont encore présents dans l'image
  - si on ne peut pas se rendre compte s'il y a un message, on peut toutefois modifier les données afin de rendre le message caché irrécupérable...

Image d'un arbre. En ne gardant que les 2 bits les moins significatifs de chaque composante de couleur, on obtient l'image suivante (après éclaircissement)

Image d'un chat extraite de l'image précédente [wikipedia]

Informatique légale (*computer forensics*)

- Techniques qui permettent de collecter, conserver et analyser des preuves issues de supports numériques en vue de les produire dans le cadre d'une action en justice
- Nous laissons énormément de traces lorsqu'on travaille sur un ordinateur :
  - sur le réseau : notre adresse IP, sites visités
  - sur l'ordinateur : fichiers temporaires
- Amazon Echo, "Arkansas judge drops murder charge in Amazon Echo case" [cnn]
- iPhone, attaque terroriste de San Bernardino, CA, USA, 12/2015, le téléphone du terroriste a été trouvé, mais Apple a refusé de le débloquer ; FBI a fait appel à un hacker pour le débloquer (vulnérabilité 0-day) ou à une compagnie israélienne

Informatique légale

- Quand on efface un fichier, son entrée dans le répertoire est effacée, mais son contenu reste là
  - effacement sécurisé de fichiers : wipe, secure-delete etc.
- apt-cache show bleachbit :
  - BleachBit deletes unnecessary files to free valuable disk space, maintain privacy, and remove junk. It removes cache, Internet history, temporary files, cookies, and broken shortcuts.
  - It handles cleaning of Adobe Reader, Bash, Beagle, Epiphany, Firefox, Flash, GIMP, Google Earth, Java, KDE, OpenOffice.org, Opera, RealPlayer, rpmbuild, Second Life Viewer, VIM, XChat, and more.
  - Beyond simply erasing junk files, BleachBit wipes free disk space (to hide previously deleted files for privacy and to improve compression of images), vacuums Firefox databases (to improve performance without deleting data), and securely shreds arbitrary files.

Conclusion

- La sécurité apparaît à tous les niveaux : matériel, logiciel, réseau, utilisateur même
- On attaque les données (lire, effacer, modifier), le matériel (l'endommager), on porte atteinte à notre vie privée (méls), on nous demande de l'argent (ransomware) ou on le vole (CB)
- Le nombre d'appareils vulnérables ne cesse d'augmenter
- De nouvelles failles de sécurités apparaissent tous les jours
- Les failles sont très variées (buffer overflow, non-initialisation, non-vérification de paramètres etc.)
- Penser toujours à la sécurité lors du développement d'un produit
- Éviter **toutes** les failles est pratiquement impossible, et on n'a jamais réussi à trouver une méthodologie qui mène à 0-vulnérabilité => on est toujours à la merci des attaquants persévérants...