Cours 6 : la sécurité dans les réseaux

- 6.1 Propriétés de sécurité
- 6.2 Cryptage des données (DES)
- 6.3 Cryptage à clé publique
- 6.4 Authentification
- 6.5 Intégrité
- 6.6 Controle d'accès (pare-feux)
- 6.7 Attaques et contre-mesures
- 6.8 Applications sécurisées
- 6.9 Un peu de virologie

1

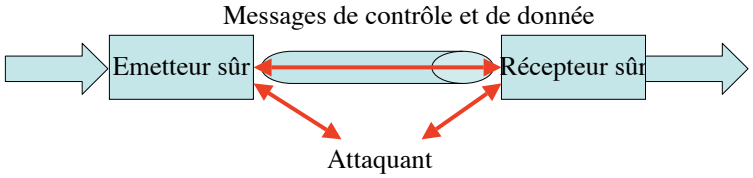
Une urbanisation numérique fragile (Internet/sans fil, "ubiquité")

- Les enjeux :
  - La maîtrise du cycle de vie des patrimoines numériques (transport, traitement, stockage) : question de souveraineté -> technique maîtresse du cryptage
  - La valorisation des contenus : enjeu économique -> tatouage, …
  - La confiance dans l'univers (république) numérique : enjeu social -> authentification, certificats, tiers de confiance, …
  - La sécurisation des infosphères :
    - Niveau individuel : préservation de l'intimité (filature électronique)
    - Niveau organisation : prévention des attaques, architectures de sécurité
    - Niveau état : invulnérabilité des infrastructures critiques, prévention des catastrophes

2

6.1 Propriétés de sécurité (les objectifs)

- Confidentialité : entre un émetteur et un récepteur. Nécessite l'encryptage de la donnée par l'émetteur et son décryptage par le récepteur
- Authentification : l'émetteur et le récepteur doivent s'assurer de l'identité du partenaire
- Intégrité et non répudiation : le message transmis n'est pas altéré
- Disponibilité et contrôle d'accès : l'attaquant ne peut empêcher l'accès aux ressources



3

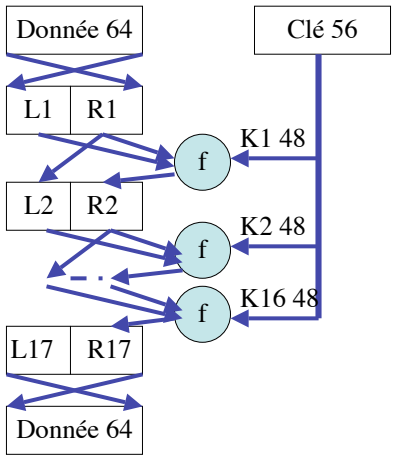
Statistiques sur la sécurité

- 20000 attaques réussies par mois sur un ou plusieurs sites dans le monde
- 70000 virus en 2004 (augmentation d'environ 1000 par mois, 10000 actifs, exemple d'un pic infection : 10% du trafic mail)
- 20 milliards de messages SPAM par jour
- Cybercriminalité évaluée à 50 milliards d'euros par an
- Tolérance étonnante des utilisateurs

4

6.2 Cryptage des données
("Data Encryption Standard" DES 1993)

- Encode des textes de 8 octets (contenant chacun un bit de parité) en utilisant une clé de 56 bits
- Fait en sorte que chaque bit de donnée dépendent de tous les autres et de la clé



5

Le "défi" DES

- Décryptage de *"Strong cryptography makes the world a safer place"* en 4 mois par des volontaires recrutés sur Internet qui ont testé 18×10^{15} clés
- En 1999 : 22 heures avec un réseau de 100000 machines
- Multiple DES, AES (jusqu'à des clés de 256 bits, chaque tour est constitué d'une substitution non linéaire, d'une permutation linéaire et de l'addition de la clé) : chiffrement en 2Gbit/s sur ASIC

6

6.3 Cryptage à clé publique

- Non seulement pour le cryptage, mais aussi pour l'authentification et la signature électronique
- Fonctions à sens unique
- Chaque utilisateur A possède 2 clés : une publique (K_A^+) et une privée (K_A^-)
- On peut trouver des clés sûres K_A^+ et K_A^- telles que : $K_A^-(K_A^+(m)) = K_A^+(K_A^-(m)) = m$
  - A!B : $K_B^+(m)$
  - B?A : $m=K_B^-(K_B^+(m))$
- Algorithme RSA (Rivest, Shamir, Adelman)

7

Algorithme RSA

- Pour recevoir un message crypté, B doit :
  - Choisir 2 entiers premiers p et q
  - Calculer n = pq
  - Choisir e < n premier avec (p-1) et (q-1)
  - Soit d tel que ed=1 mod (p-1)(q-1) (théorème de Bachet de Méziriac)
  - $K_B^+ = (n,e)$, $K_B^- = (n,d)$
- Pour envoyer une suite de bits représentée par un nombre m < n, A envoie $c = m^e \bmod n$
- Pour décrypter, B calcule $m = c^d \bmod n$
- En pratique n est représenté sur 1024 bits

8

Petit exemple

- p = 5, q = 7
- n = 35, (p-1)(q-1) = 24
- e = 5, d = 29

Texte	Rep. Num	m^e	Code $c = m^e \bmod n$	c^d	$m = c^d \bmod n$	Texte
E	5	3125	10	10000000000000000000000000000000	5	E
N	14	537824	14	172867373967747110156721694598758 4	14	N
S	19	2476099	24	106200365064067167761572429136211990282 24	19	S

L'exponentiation est coûteuse. RSA est plus sûr mais beaucoup moins rapide que DES[9]

La théorie

- Cryptage/décryptage calcule :
  $(m^e \bmod n)^d = (m^e)^d \bmod n = m^{ed} \bmod n$
- Théorème (Euler) : $x^{(p-1)(q-1)} = 1 \bmod n$
- $m^{ed} \bmod n = m^{(1+k(p-1)(q-1))} \bmod n$
  $= m.(m^{(p-1)(q-1)})^k \bmod n = m$
- La sécurité est fondée sur la non connaissance d'algorithme rapide de factorisation (non garantie)

10

6.4 Authentification

- Preuve en ligne de l'identité
- Il ne suffit pas de donner son identité en clair, car un intrus peut se faire passer pour un autre, y compris en cas d'envoi de mot de passe (qui peut être observé)
- Le cryptage du mot de passe n'est pas une solution (l'intrus peut rejouer le scénario) -> mots de passe à usage unique : notion de "nonce"

11

Exemple de protocole cryptographique

- A!B : je suis A
- B!A : nonce R
- A!B : $K_A^-(R)$
- B!A : demande K_A^+
- A!B : K_A^+
- B calcule $K_A^+(K_A^-(R)) = R$, authentifiant A

12

Attaque

- Intrus!B : je suis *A*
- B!Intrus : nonce R
- Intrus!B : $K_{Intrus}^{-}(R)$
- B!Intrus : demande K_A^{+}
- Intrus!B : K_{Intrus}^{+}
- B calcule $K_{Intrus}^{+}(K_{Intrus}^{-}(R)) = R$, authentifiant Intrus comme *A* !

Mais A pourrait se rendre compte de cette attaque

13

Attaque par interception

- A!B : je suis *A*
- Intrus!B : je suis *A*
- B!Intrus : nonce R
- Intrus!B : $K_{Intrus}^{-}(R)$
- B!Intrus : demande K_A^{+}
- Intrus!B : K_{Intrus}^{+}
- Intrus!*A* : R
- *A*!Intrus : $K_A^{-}(R)$
- Intrus!*A* : demande K_A^{+}
- *A*!Intrus : K_A^{+}
- B!Intrus : $K_{Intrus}^{+}(X)$ /* donnée cryptée X */
- Intrus décrypte $X = K_{Intrus}^{-}(K_{Intrus}^{+}(X))$
- Intrus!*A* : $K_A^{+}(X)$
- *A* décrypte $X = K_A^{-}(K_A^{+}(X))$

A et B ne se rendent compte de rien…

14

Protocole Needham-Schroeder, 1978

S est un serveur de clés publiques

- A!S : B
- S!A : $K_S^-(K_B^+,B)$
- A!B : $K_B^+(R_A,A)$
- B décrypte $(R_A,A) = K_B^-(K_B^+(R_A,A))$
- B!S : A
- S!B : $K_S^-(K_A^+,A)$
- B!A : $K_A^+(R_A,R_B)$
- A décrypte $(R_A,R_B) = K_A^-(K_A^+(R_A,R_B))$ /* Retrouver R_A authentifie B */
- A!B : $K_B^+(R_B)$
- B décrypte $R_B = K_B^-(K_B^+(R_B))$ /* Retrouver R_B authentifie A */

15

Attaque de Lowe, 1995

1. A!S : I /* Le dialogue vers I sera utilisé pour usurper l'id A */
2. S!A : $K_S^-(K_I^+,I)$
3. A!I : $K_I^+(R_A,A)$
4. I!S : B
5. S!I : $K_S^-(K_B^+,B)$
6. I_A!B : $K_B^+(R_A,A)$ /* I se fait passer pour A */
7. B décrypte $(R_A,A) = K_B^-(K_B^+(R_A,A))$
8. B!S : A
9. S!B : $K_S^-(K_A^+,A)$
10. B!A : $K_A^+(R_A,R_B)$ /* I intercepte : Correction : $K_A^+(R_A,R_B,B)$ */
11. I!A : $K_A^+(R_A,R_B)$
12. A décrypte $(R_A,R_B) = K_A^-(K_A^+(R_A,R_B))$
13. A!I : $K_I^+(R_B)$
14. I_A!B : $K_B^+(R_B)$ /* I se fait passer pour A */

Attaque avec confusion nom et nonce, 1996

16

6.5 Intégrité

- Signature électronique ("off line") :
  1. A!B : $K_A^-(m)$
  2. B décrypte $K_A^+(K_A^-(m)) = m$ pour vérifier que le document m a bien été signé par A
- Empreintes : utiliser un résumé (analogue à un checksum) H(m)
  - La signature est $K_A^-(H(m))$
  - H est une fonction de hachage telle qu'il est très difficile de trouver deux messages x et y tels que H(x) = H(y)
- Algorithme MD5 : résumé de 128 bits (conjecture : trouver un message d'un résumé donné demanderait 2^{128} opérations)
- SHA-1 (Secure Hash Algorithm) : 160 bits (US federal standard)

17

6.6 Controle d'accès

- Un pare-feu est un serveur qui s'intercale entre le réseau administré et le réseau extérieur (il peut être au niveau d'une organisation ou même installé sur votre machine)
  - Filtrage des paquets (sur les adresses IP ou les ports TCP, sur le type des messages : TCP Syn, TCP ack, messages ICMP). Par exemple, filtrage de Telnet pour interdire les connexions distantes, ou filtrage d'UDP pour interdire les applications des vendeurs audio-vidéo offrant un mode UDP par défaut. Utilisation de listes noires. Utilisation du bit ack pour rendre dissymétrique la connexion (utilisation de serveurs externes depuis l'intérieur, mais interdiction d'utiliser des serveurs internes depuis l'extérieur) -> besoin de formaliser une "politique de sécurité" (actuellement mis en œuvre par un ensemble de règles)
  - Passerelle d'application : exemple de la machine transit de l'Irisa (seules les connexions telnet de cette machine ne sont pas filtrées). Cette application peut dialoguer avec l'utilisateur pour l'authentifier. L'inconvénient est qu'elle dépend de l'application considérée. Preuves de sécurité des serveurs ?

18

6.7 Attaques et contre-mesures

- Scrutation ("mapping"): "ping" peut être utilisé pour trouver des adresses IP (celles qui répondent). Idem au niveau TCP sur les ports. -> les pare-feux peuvent repérer les comportements de scrutation.
- Espionnage des paquets ("packet sniffing") : facile à l'intérieur d'un réseau local Ethernet. Les trames espionnées sont passées à un programme décodeur pour récupérer l'information applicative pertinente (mots de passe par exemple) : exemple du FBI Carnivore à travers l'ensemble du réseau. -> détection des machines utilisant le mode d'observation (adresse IP de diffusion). -> cryptage systématique.
- Usurpation d'identité ("spoofing") : changer l'adresse source IP -> demander aux routeurs de vérifier que les adresses entrantes sont correctes.

19

- Dénis de service ("Denial of service attacks") : rendre le réseau inutilisable. Exemples : 1/ inondation de demandes de connexions TCP avec des adresses usurpées. 2/ envoi de fragments sans jamais compléter le paquet. 3/ envoi d'enquêtes ICMP pour un destinataire usurpé.
- Dénis par attaques réparties : installation clandestine d'attaquants et déclenchement synchronisé. -> difficile à contourner. -> vers une plus grande traçabilité des échanges.
- Prise d'otage ("Hijacking") : un intrus s'interpose entre les partenaires

20

6.8 Applications sécurisées

- Messagerie cryptée : PGP ("Pretty Good Privacy")
- SSL ("Secure Sockets Layer") et TLS ("Transport Layer Security")
- Ipsec : une pile IP pour la sécurité
  - Authentification Header Protocol (AH)
  - Encapsulation Security Payload Protocol (ESP)
- Sécurité radio : espionnage autour des batiments -> protocole WEP ("Wired Equivalent Privacy").

21

6.9 Un peu de virologie
[Eric Filiol, Springer/IRIS]

- Auto-reproduction :
  - Autoprint :

```
p="p=%c%s%c;main(){printf(p,34,p,34);}";main(){printf(p,34,p,34);}
```

  - Automates cellulaires (constructeurs universels) :
    Von Neumann (1948), Codd (1968), Langton (1984), Byl (1989), Ludwig (1993)
- Notion de virus (fin 1970 (officiellement))
  Thèse Fred Cohen (1986), directeur Léonard Adleman (US)

22

Table de transitions de Byl2 (CHDBG-N)

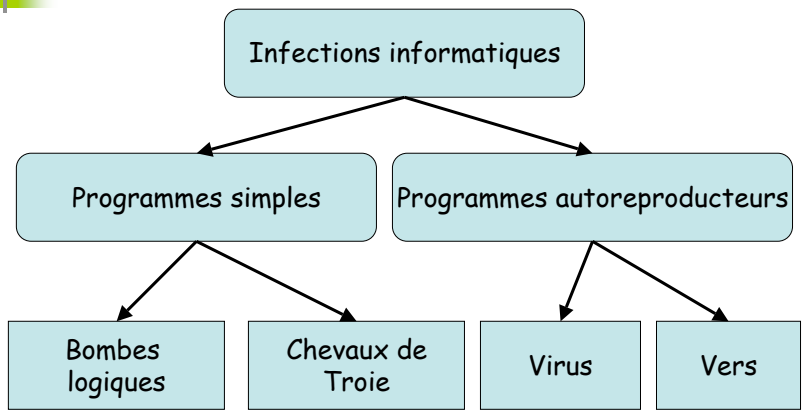
- 00003-1/00012-2/00013-1/00015-2/00025-5/00031-5/00032-3/00042-2/0****-0
- 10000-0/10001-0/10003-3/10004-0/10033-0/10043-1/10321-3/11253-1/12453-3/1****-4
- 20000-0/20015-5/20022-0/20202-0/20215-5/20235-3/20252-5/2****-2
- 30001-0/30003-0/30011-0/30012-1/30121-1/30123-1/31122-1/31123-1/31215-1/31223-1/31233-1/31235-5/31432-1/31452-5/3****-3
- 40003-5/40043-4/40212-4/40232-4/40242-4/40252-0/40325-5/4****-3
- 50022-5/50032-5/50212-4/50222-0/50322-0/5****-2

Une séquence reproductrice

22	22	
2312	2342	...
2342	2332	
25	22	

Duplication en 25 pas

Taxonomie



25

Principes de l'infection

- Le programme infectant est porté par un programme hôte ("dropper")
- Lorsque le dropper est exécuté :
  - Le programme infectant prend la main
  - Puis il la rend au programme hôte sans trahir sa présence
- Scénario d'attaque :
  - Recherche des programmes cibles (fichiers exécutables, évitement de la sur-infection (signature exploitable))
  - Copie du code dans la cible
  - Programme d'anti-détection
  - Charge finale (différée ou non)

26

Infections simples

- Bombes logiques (souvent les charges finales). Souvent différées (exemple du programmeur système)
- Chevaux de Troie ("trojans") :
  - Sa partie serveur est installée à l'insu de la victime (téléchargement à partie de sites leurres par exemple)
  - Celle-ci donne discrètement à l'attaquant l'accès à des ressources
  - Le client (attaquant) recherche les machines infectées sur le réseau et en prend le contrôle
- Leurres (fausses bannières par exemple)

27

Modes d'action des virus

- Par écrasement de code :
  - Entête : le code infecté devient non exécutable
  - Ailleurs : il faut insérer une instruction de saut vers le début du virus (sinon, permet un semblant de furtivité, l'erreur se déclenchant après un début d'exécution)
  - Remplacement (détectable car les fichiers infectés ont tous la même taille)
- Par recouvrement de code :
  - Prepend : difficile à cause des adresses à recalculer
  - Append : insertion d'un saut initial, puis restauration

28

- Par entrelacement de code :
  - Utilise le format PE des exécutables Windows dans lequel le code est fragmenté par plages fixes non nécessairement remplies. Le virus peut se glisser dans les espaces libres (la taille de l'exécutable infecté reste inchangée)
- Par accompagnement de code :
  - Le code viral identifie une cible et duplique son code en créant un fichier supplémentaire
  - Lorsque l'utilisateur exécute le programme cible, la copie virale est exécutée en premier (propagation) :
    - Exécution préemptive (sous DOS, f.com -> f.exe -> f.bat)
    - Utilisation du PATH
    - Renommage et sauvegarde du fichier cible
- Virus de code source (stdio.h par exemple)

29

Techniques anti-antivirales

- Furtivité :
  - On cache le virus dans des zones déclarées faussement défectueuses par exemple
  - Désinfection après activation
- Polymorphisme :
  - Le virus se réécrit sous une forme différente (changement de signature)
  - Chiffrement : le début du code en clair est de déchiffrer la suite (la procédure de chiffrement peut aussi être changée à chaque infection)
- Attaque des anti-viraux

30

Techniques antivirales

- Recherche de signatures (base de signatures à mettre à jour)
- Analyse spectrale : indice de programmation "déviante" par rapport à ce que produit un compilateur
- Règles heuristiques
- Contrôle d'intégrité des fichiers
- Surveillance comportementale
- Emulation du code dans une zone confinée

31

32