Sécurisation des protocoles

Mauvais exemples

Carlos Aguilar
carlos.aguilar-melchor@isae-supaero.fr
Benoît Morgan
benoit.morgan@enseeiht.fr

IRIT

Sources

[1] https://tools.ietf.org/html/rfc1094

[2] https://tools.ietf.org/html/rfc1813

[3] https://tools.ietf.org/html/rfc5531

[4] https://tools.ietf.org/html/rfc1014

[5] https://en.wikipedia.org/wiki/X_Window_authorization

[5] https://tools.ietf.org/html/rfc1013

Plan

Objectifs du cours

- Étudier de nouveaux mais vieux protocoles

Objectifs du cours

- Étudier de nouveaux mais vieux protocoles
- Comprendre le contexte et hypothèse de l'époque

Objectifs du cours

- Étudier de nouveaux mais vieux protocoles
- Comprendre le contexte et hypothèse de l'époque
- Comprendre pourquoi certains choix ont été fait

Objectifs du cours

- Étudier de nouveaux mais vieux protocoles
- Comprendre le contexte et hypothèse de l'époque
- Comprendre pourquoi certains choix ont été fait
- Comprendre "ce qu'il ne faut pas faire"

Contexte

Réseau

- Inter laboratoire et universités
- Pas de particuliers
- Très peu de membres

Contexte

Réseau

- Inter laboratoire et universités
- Pas de particuliers
- Très peu de membres

Conception des protocoles

- Efforts sur l'aspect service
- Un peu de sûreté de fonctionnement

Contexte : sûreté de fonctionnement vs sécurité

Modèle des pairs et moyens de communication

- Honnêtes
- Tolérance aux fautes / suppression des fautes
- Sujets aux erreurs et défaillances

Contexte : sûreté de fonctionnement vs sécurité

Modèle des pairs et moyens de communication

- Honnêtes
- Tolérance aux fautes / suppression des fautes
- Sujets aux erreurs et défaillances

Sécurité ?

- Pas de prise en compte des malveillances
- Confiance aux systèmes pour l'authentification et l'autorisation des utilisateurs
- Confiance mutuelle en systèmes

Contexte : sûreté de fonctionnement vs sécurité

Modèle des pairs et moyens de communication

- Honnêtes
- Tolérance aux fautes / suppression des fautes
- Sujets aux erreurs et défaillances

Sécurité ?

- Pas de prise en compte des malveillances
- Confiance aux systèmes pour l'authentification et l'autorisation des utilisateurs
- Confiance mutuelle en systèmes

Conséquences ?

- Globalement *open bar*
- Si corruption d'un système
- Ou accès au médium

Protocoles classique

Protocoles et outils courants dans le monde Unix

- Protocoles en “r” : rlogin, rsh, rcp, etc.
- FTP (transfert de fichiers)
- NFS (partage de fichiers)
- X : sessions graphiques

Flux en clair

- Données en clair
- Mots de passe en clair (telnet, FTP)

TFTP

Utilisé par de nombreux petits dispositifs hardware ...
Récupération/modification des fichiers de configuration

Protocoles en r

Berkeley **r-commands** : services

- Ensemble de programmes UNIX (seulement)
- Se connecter et obtenir un terminal virtuel à distance
- Exécuter des actions à distance

Protocoles en r

Berkeley **r-commands** : services

- Ensemble de programmes UNIX (seulement)
- Se connecter et obtenir un terminal virtuel à distance
- Exécuter des actions à distance

Quelques programmes principaux

Prog.	**Description**	*Login*
`rlogin`	Connexion pour obtenir un terminal virtuel	oui
`rsh`	Exécute un programme dans un *shell*	non
`rexec`	Idem `rsh`	oui
`rcp`	Copie de fichier entre machines `rcp file.txt host:file.txt`	non
`rwho`	`who` distant	non
`rstat`	`stat` distant	non
`ruptime`	`uptime` distant	non

Protocoles en r

Notion de relation de confiance

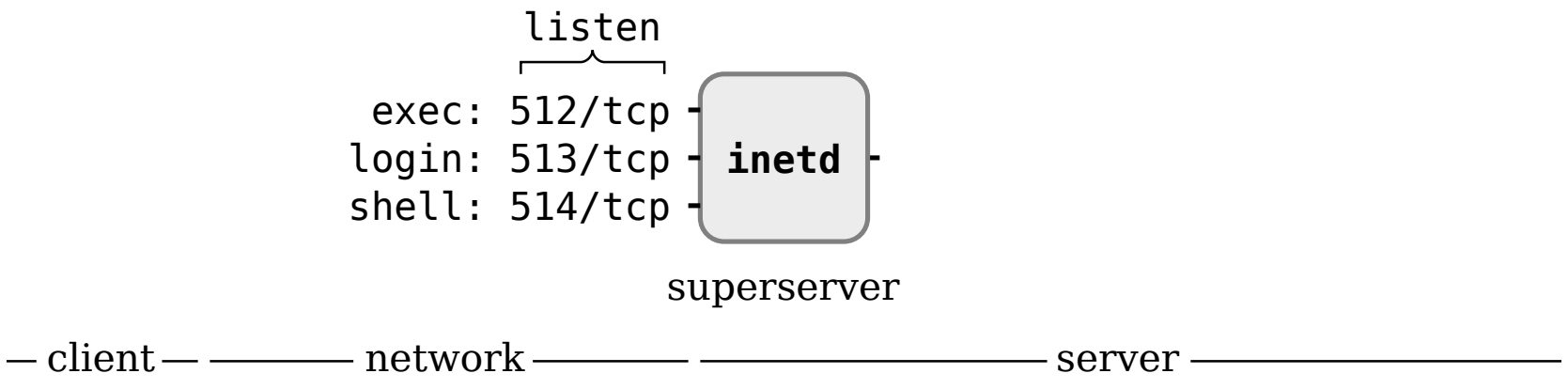
- `/etc/rhosts` ou `/etc/hosts.equiv` et `~/.rhosts`
- Contenu : lignes sous la forme `hôte utilisateur` (e.g. `+ +`)

Permet à des utilisateurs authentifiés auprès de `hôte` d'accéder aux ressources locales

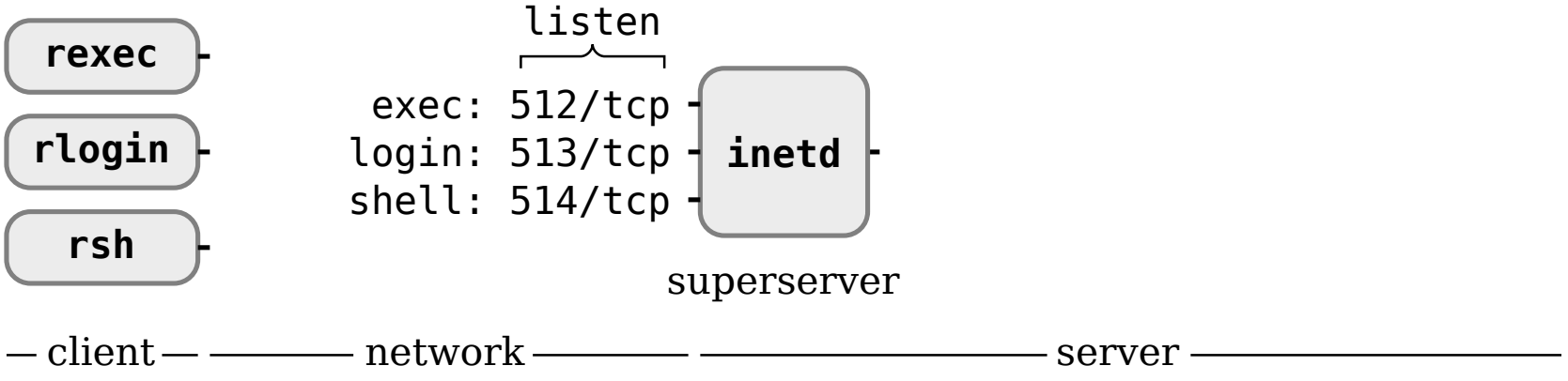
Exemples

- N'importe qui d'authentifié sur pikachu peut utiliser carapuce avec n'importe quel compte
- N'importe qui pouvant usurper pikachu peut utiliser carapuce avec n'importe quel compte ...

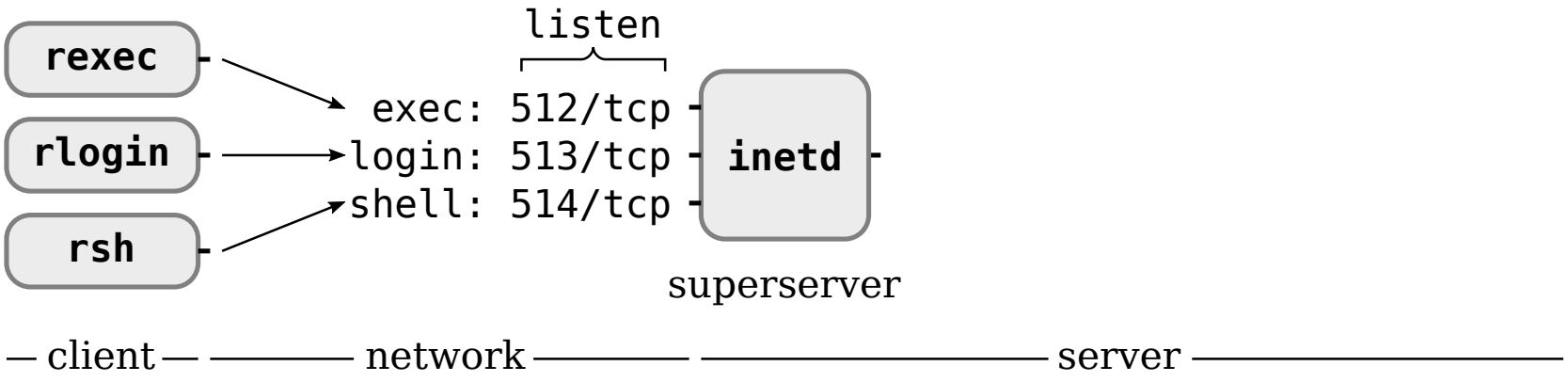
Infrastructure pour protocoles en r



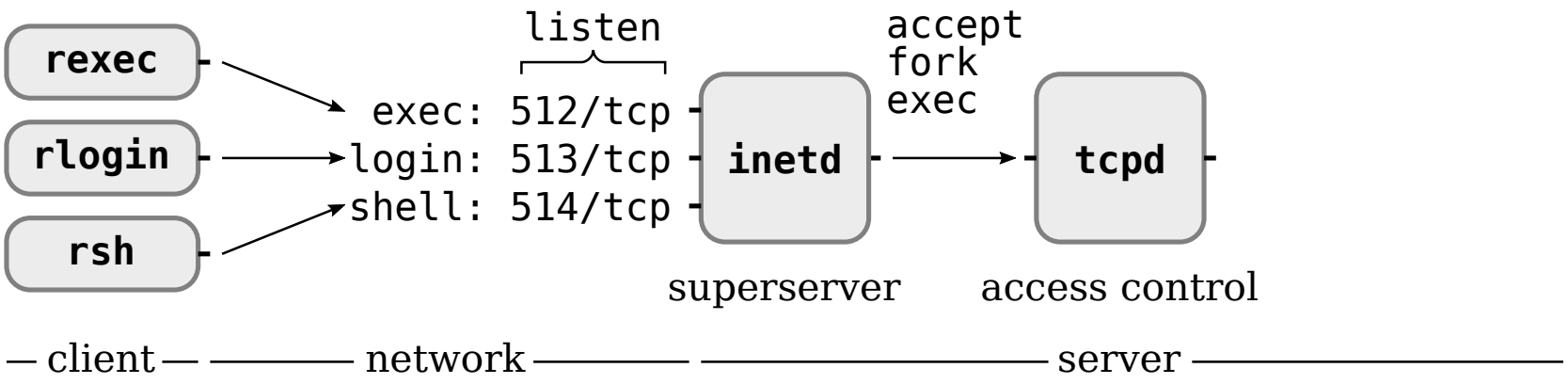
Infrastructure pour protocoles en r



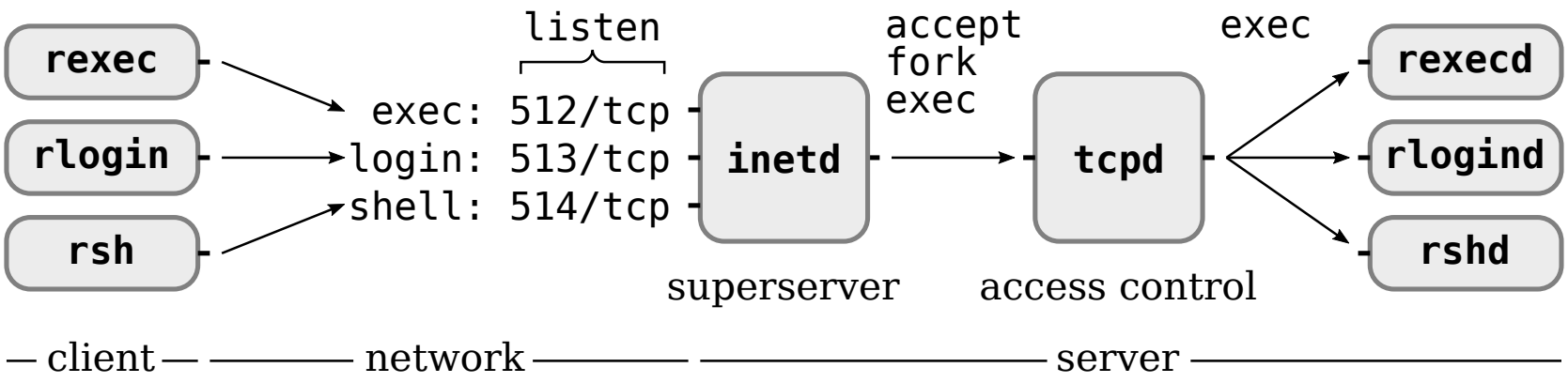
Infrastructure pour protocoles en r



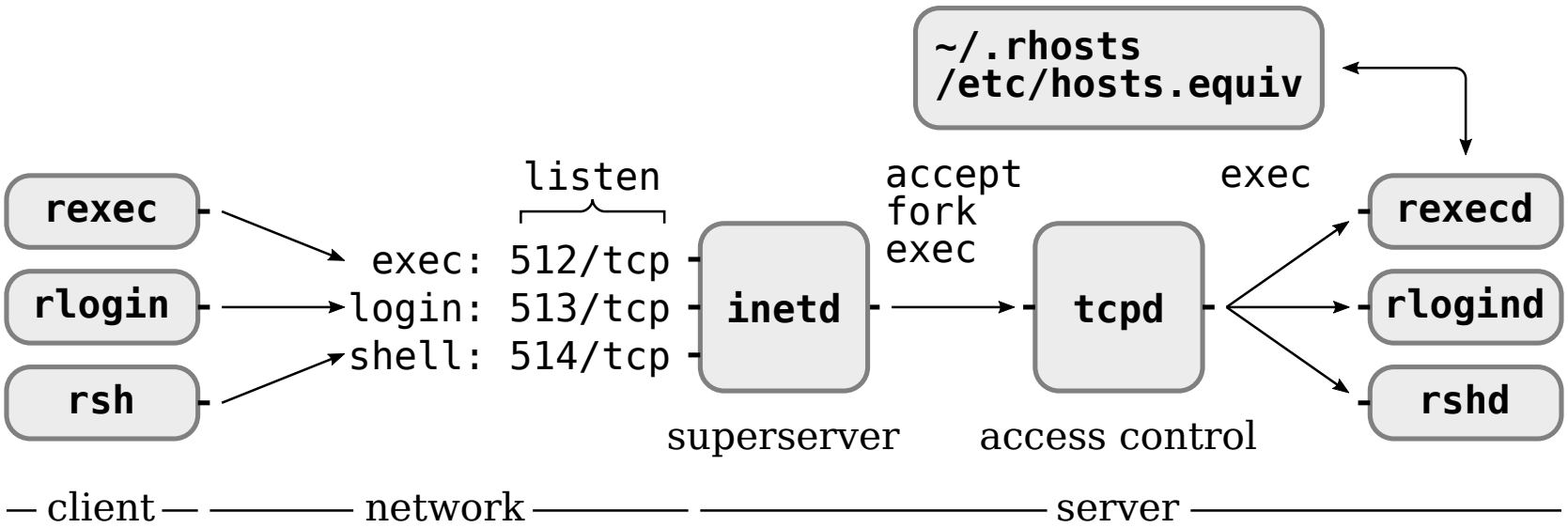
Infrastructure pour protocoles en r



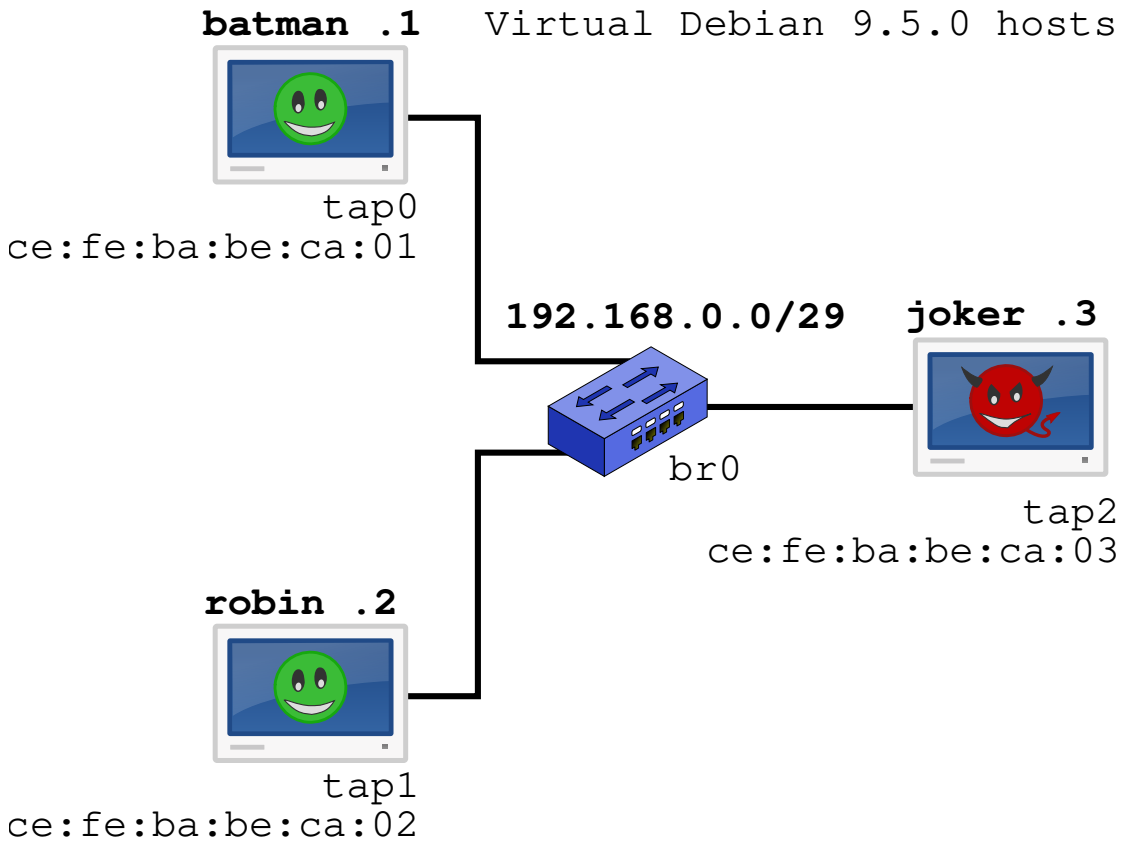
Infrastructure pour protocoles en r



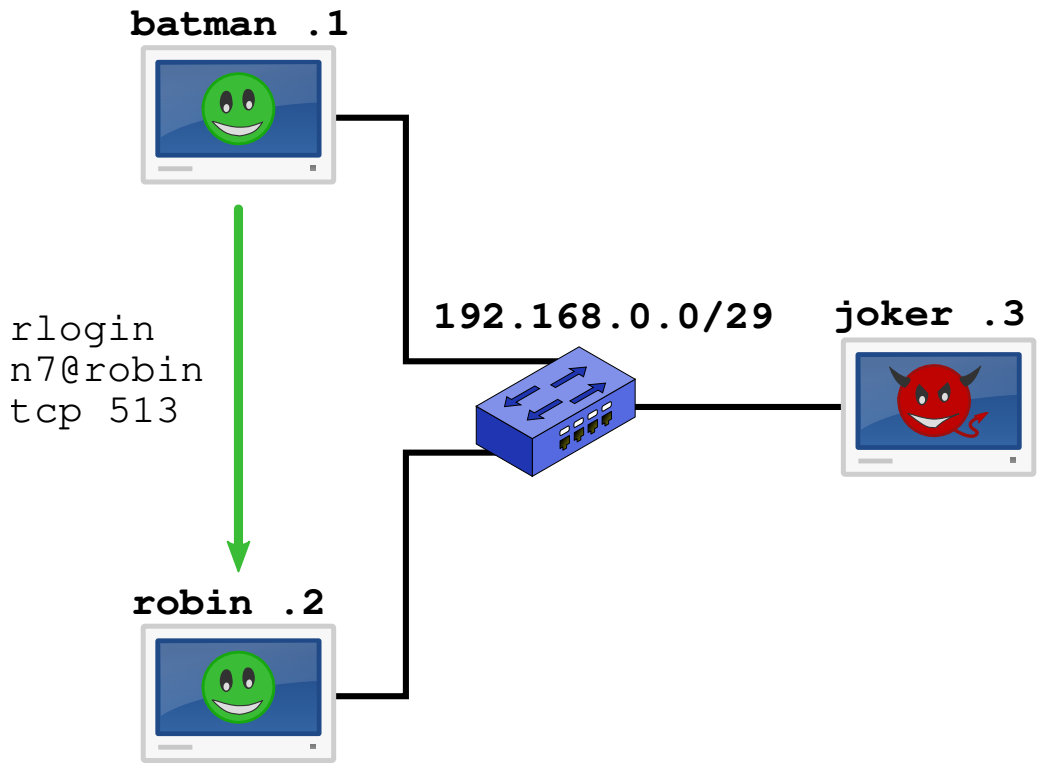
Infrastructure pour protocoles en r



Démo : protocoles en r



Démo : `rlogin`



Interaction normale : utilisation de `rlogin` de batman sur robin

Démo : `rlogin`

Ouverture de terminal virtuel sur robin depuis batman

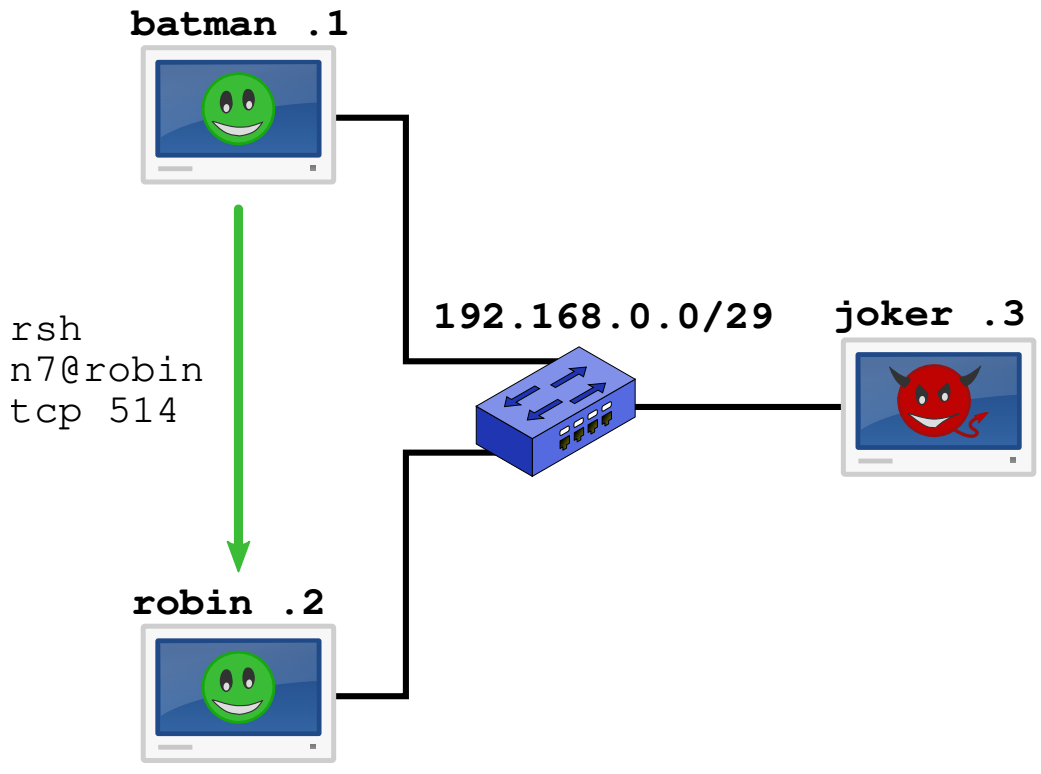
```
n7@batman:~$ rlogin -l n7 robin
[...MOTD...]
n7@robin:~$ ls
n7@robin:~$ ls # Le home est vide :)
```

Démo : `rlogin`

```
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Start Handshake
192.168.0.1 192.168.0.2 Rlogin       84 User name: n7, Data: n7\000n7\000linux/38400\000
192.168.0.2 192.168.0.1 Rlogin       67 User name: n7, Startup info received
192.168.0.2 192.168.0.1 Rlogin       76 User name: n7, Data: Password:
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Data: s
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Data: n
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Data: \r
192.168.0.2 192.168.0.1 Rlogin       68 User name: n7, Data: \n\r
192.168.0.2 192.168.0.1 Rlogin       67 User name: n7, Control Message (Window size request)
192.168.0.1 192.168.0.2 Rlogin       78 User name: n7, Terminal Info (rows=48, cols=128)
192.168.0.2 192.168.0.1 Rlogin      114 User name: n7, Data: Last login: Tue Oct 30 10:11:41 CET 2018 on tty1
192.168.0.2 192.168.0.1 Rlogin      439 User name: n7, Data: \r\nLinux robin 4.9.0-7-amd64 #1 SMP Debian 4.9.110-3+deb9u2
192.168.0.2 192.168.0.1 Rlogin       78 User name: n7, Data: n7@robin:~$
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Data: l
192.168.0.2 192.168.0.1 Rlogin       67 User name: n7, Data: l
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Data: s
192.168.0.2 192.168.0.1 Rlogin       67 User name: n7, Data: s
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Data: \r
192.168.0.2 192.168.0.1 Rlogin       68 User name: n7, Data: \r\n
192.168.0.2 192.168.0.1 Rlogin       78 User name: n7, Data: n7@robin:~$
192.168.0.1 192.168.0.2 Rlogin       67 User name: n7, Data: \004
192.168.0.2 192.168.0.1 Rlogin       80 User name: n7, Data: d\303\251connexion\r\n
```

Confidentialité du lien ?

Démo : `rsh`



Interaction normale : utilisation de `rsh` de batman sur robin pour exécuter `ls /`

Démo : `rsh`

Exécution de ls dans un shell sur robin depuis batman

```
n7@batman:~$ rsh robin ls /
Persmission denied.
n7@batman:~$
```

Ajout de l'utilisateur n7 et de l'hôte batman comme hôte de confiance sur robin

```
n7@robin~# echo "batman n7" >> /etc/hosts.equiv
```

Et rebelotte !

Démo : rsh

Exécution de ls dans un shell sur robin depuis batman

```
n7@batman:~$ rsh robin ls /
bin
boot
dev
etc
home
initrd.img
initrd.img.old
[...]
n7@batman:~$
```

Démo : `rsh`

Échange protocolaire

```
192.168.0.1  192.168.0.2   RSH          71 Session Establishment
192.168.0.1  192.168.0.2   RSH          77 Session Establishment
192.168.0.2  192.168.0.1   RSH          67 Server username:n7 Server -> Client Data
192.168.0.2  192.168.0.1   RSH         211 Server username:n7 Server -> Client Data
```

Échange de l'identité

```
▼ Remote Shell
    Client username: n7
    Server username: n7
    Command to execute: ls /
```

Établissement de session

```
▼ Remote Shell
    Stderr port (optional): 1022
```

Démo : `rsh`

Échange protocolaire

```
192.168.0.1  192.168.0.2   RSH          71 Session Establishment
192.168.0.1  192.168.0.2   RSH          77 Session Establishment
192.168.0.2  192.168.0.1   RSH          67 Server username:n7 Server -> Client Data
192.168.0.2  192.168.0.1   RSH         211 Server username:n7 Server -> Client Data
```

Échange de l'identité

```
▼ Remote Shell
    Client username: n7
    Server username: n7
    Command to execute: ls /
```

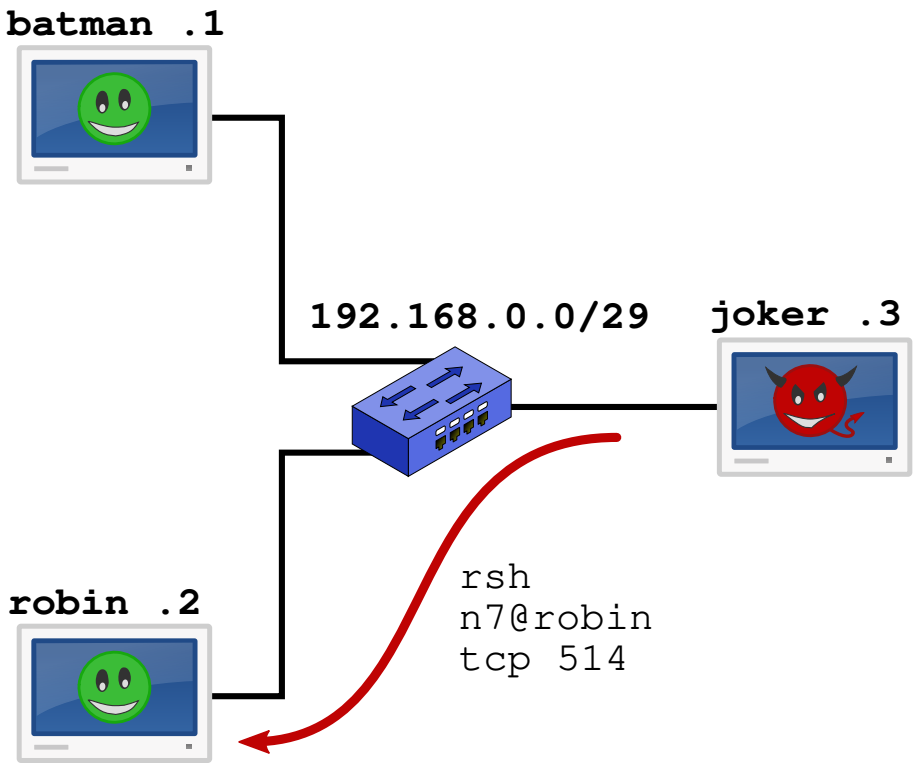
Établissement de session

```
▼ Remote Shell
    Stderr port (optional): 1022
```

Sécurité ?

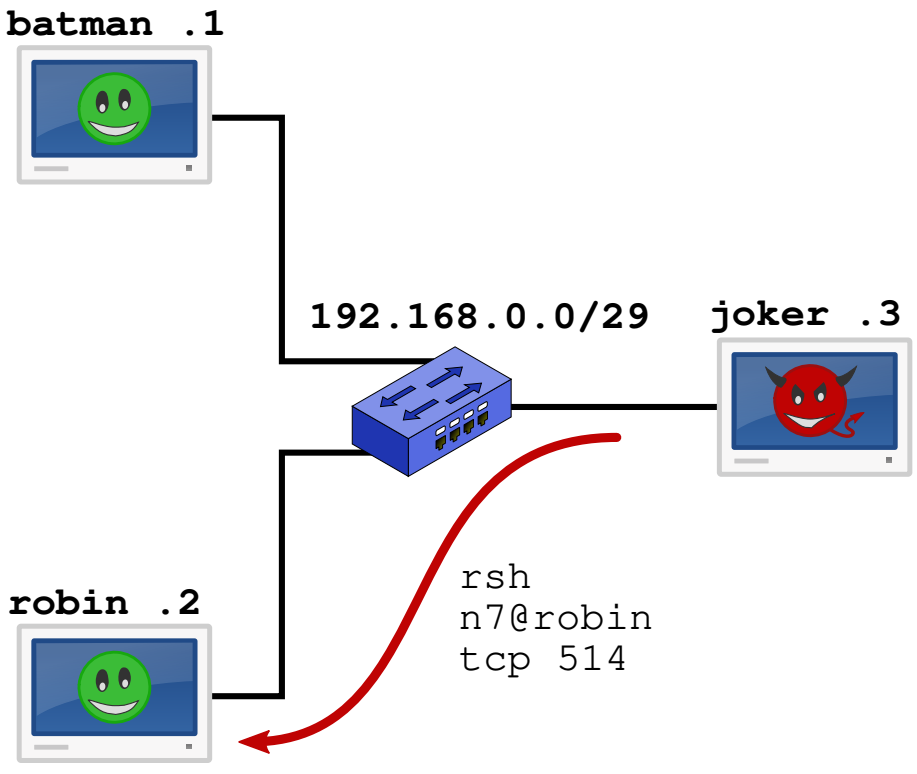
- Intégrité des *username* ?
- Intégrité de l'adresse IP source ?

Démo : attaque de `rsh`



Interaction malveillante : utilisation de `rsh` de joker sur robin
pour exécuter `touch /home/n7/p0wnd`
en usurpant l'identité de `n7@batman`

Démo : attaque de `rsh`



Interaction malveillante : utilisation de `rsh` de joker sur robin
pour exécuter `touch /home/n7/p0wnd`
en usurpant l'identité de `n7@batman`
Conseil : Attaque ARP et usurpation d'adresse IP ? :)

Démo : attaque de `rsh`

Démo !

```
n7@batman:~$ echo 'robin n7' | sudo tee /etc/hosts.equiv
n7@batman:~$ exit
n7@joker~$ /rw/arp-poisonning-attack.sh
n7@joker~$ /rw/rsh-alias-attack.sh
n7@batman:~$ dir # :)
n7@batman:~$ ls
n7@batman:~$ dir
n7@batman:~$ alias ls # :)
n7@batman:~$ unalias ls
n7@batman:~$ ls
```

NFS : partage de fichiers

Concept

- Partage réseau de fichiers pour sustèmes UNIX
- Transparence de la gestion des accès

$\Rightarrow$ Nécessité de cohérence des identifiants d'utilisateurs entre machines

$\Rightarrow$ S'intègre assez bien avec un annuaire d'entreprise (`ldap`)

NFS : partage de fichiers

Concept

- Partage réseau de fichiers pour sustèmes UNIX
- Transparence de la gestion des accès

⇒ Nécessité de cohérence des identifiants d'utilisateurs entre machines

⇒ S'intègre assez bien avec un annuaire d'entreprise (`ldap`)

Mount access

Donné à des adresses IP par répertoire partagé

NFS : partage de fichiers

Concept

- Partage réseau de fichiers pour sustèmes UNIX
- Transparence de la gestion des accès

⇒ Nécessité de cohérence des identifiants d'utilisateurs entre machines

⇒ S'intègre assez bien avec un annuaire d'entreprise (`ldap`)

Mount access

Donné à des adresses IP par répertoire partagé

Mise en œuvre sous linux

- Module du noyau
- Nouveau système de fichiers
- Interfacé avec l'infrastructure linux
  - `# mount -t nfs [-o nfsvers=3] host:/dir /mountpoint`
  - `# umount /mountpoint`

NFS : administration sous Debian 9 *stretch*

Installation

- Version mode noyau
- Supporte uniquement les version 3 et 4 du protocole [1], [2]
- Paquet : `nfs-kernel-server`
- Activation : `# systemctl enable nfs-kernel-server`
- Démarrage : `# systemctl start nfs-kernel-server`

NFS : administration sous Debian 9 *stretch*

Installation

- Version mode noyau
- Supporte uniquement les version 3 et 4 du protocole [1], [2]
- Paquet : `nfs-kernel-server`
- Activation : `# systemctl enable nfs-kernel-server`
- Démarrage : `# systemctl start nfs-kernel-server`

Configuration (simple)

- Répertoires partagés : `/etc/exports`
- `# echo '/home batman(rw,sync,no_subtree_check,no_root_squash)' >> /etc/exports`
- `# exportfs -a`

NFS : protocole (version 3)

Généralités

- Basé sur l'appel de procédures à distance : RPC
- Données présentées à l'aide de XDR
- Transporté par UDP et TCP en fonction du programme RPC

NFS : protocole (version 3)

Généralités

- Basé sur l'appel de procédures à distance : RPC
- Données présentées à l'aide de XDR
- Transporté par UDP et TCP en fonction du programme RPC

Services du serveur

- Programme RPC MOUNT
  Sert uniquement lors de la procédure de montage
- Programme RPC NFS
  Sert pour toutes les autres interactions

NFS : protocole (version 3)

Montage de `robin:/home` depuis `batman`

1. `batman` exécute la procédure `NULL` sur NFS
   → Test des droits et de version
2. `batman` exécute la procédure `NULL` sur MOUNT
   → Test des droits et de version
3. `batman` exécute la procédure `MNT` sur MOUNT
   → retour d'un *fhandle* à utiliser avec NFS
4. `batman` exécute la procédure `FSINFO` sur NFS
   → retour d'informations sur le distème de fichier :
   taille max d'un fichier; support des liens symboliques; ...
5. `batman` exécute la procédure `PATHCONF` sur NFS
   → retour d'informations sur le dossier exporté:
   taille max du nom d'un fichier; sensibilité à la casse; ...
6. `batman` exécute la procédure `GETATTR` sur NFS
   → retour des attributs du dossier point de montage associé au *fhandle*

NFS : protocole (version 3)

Listing des fichiers `robin:/home` depuis `batman`

1. `batman` exécute la procédure `GETATTR` sur NFS
   → retour des attributs du dossier point de montage associé au *fhandle*
2. `batman` exécute la procédure `ACCESS` sur NFS
   → verification des droits d'accès associé au *fhandle* (même si le mode est transmis avec GETATTR)
3. `batman` exécute la procédure `READDIRPLUS` sur NFS
   → listing du dossier associé au *fhandle*

NFS : *Remote Procedure Call*

Généralités

- Interface de programmation associée à numéro de programme
  - `ntfs : 100003`
  - `mountd : 100005`
- Numéros uniques attribués par l'IANA
  - `$ cat /etc/rpc`
- Protocole de session indépendant du transport [2, 3]
- Sémantique de l'appel de fonction (appel, retour)

NFS : *Remote Procedure Call*

Généralités

- Interface de programmation associée à numéro de programme
  - `ntfs : 100003`
  - `mountd : 100005`
- Numéros uniques attribués par l'IANA
  - `$ cat /etc/rpc`
- Protocole de session indépendant du transport [2, 3]
- Sémantique de l'appel de fonction (appel, retour)

Lien avec la couche transport

- NFS utilise les protocoles TCP/UDP/IP
- Sémantique du numéro de service
- Point de rendez-vous ?

$\Rightarrow$ Utilisation d'un *port mapper*

NFS : *port mapper*

Service de ”rencontres” pour programmes RPC sérieux

- Point de rendez-vous connu (`tcp/111, udp/111`)
- Utilise RPC et XDR
- Fonction GETPORT : `program number` → `port number`

NFS : *port mapper*

Service de ”rencontres” pour programmes RPC sérieux

- Point de rendez-vous connu (`tcp/111, udp/111`)
- Utilise RPC et XDR
- Fonction GETPORT : `program number` → `port number`

Utilisation par NFS

- Récupération du port associé à NFS
- Récupération du port associé à MOUNT

NFS : *port mapper*

Service de ”rencontres” pour programmes RPC sérieux

- Point de rendez-vous connu (`tcp/111, udp/111`)
- Utilise RPC et XDR
- Fonction GETPORT : `program number` → `port number`

Utilisation par NFS

- Récupération du port associé à NFS
- Récupération du port associé à MOUNT

```
 4 0.022462134 192.168.0.1 192.168.0.2  Portmap    126 V2 GETPORT Call (Reply In 6) NFS(100003) V:3 TCP
 6 0.022754916 192.168.0.2 192.168.0.1  Portmap     98 V2 GETPORT Reply (Call In 4) Port:2049
21 0.023469338 192.168.0.1 192.168.0.2  Portmap     98 V2 GETPORT Call (Reply In 22) MOUNT(100005) V:3 UDP
22 0.023544004 192.168.0.2 192.168.0.1  Portmap     70 V2 GETPORT Reply (Call In 21) Port:43262
32 0.024635495 192.168.0.1 192.168.0.2  Portmap    154 V2 GETPORT Call (Reply In 34) NFS(100003) V:3 TCP
34 0.024886372 192.168.0.2 192.168.0.1  Portmap     98 V2 GETPORT Reply (Call In 32) Port:2049
```

```
▼ Portmap GETPORT Call NFS(100003) Version:3 TCP
    [Program Version: 2]
    [V2 Procedure: GETPORT (3)]
    Program: NFS (100003)
    Version: 3
    Proto: TCP (6)
    Port: 0
```

```
▼ Portmap GETPORT Reply Port:2049 Port:2049
    [Program Version: 2]
    [V2 Procedure: GETPORT (3)]
    Port: 2049
```

NFS : *eXternal Data Representation*

Généralité

- Standard de description et d'encodage des données [4]
- Utile pour transférer des données entre machines aux différentes architectures
- Langage de description des données

NFS : *eXternal Data Representation*

Généralité

- Standard de description et d'encodage des données [4]
- Utile pour transférer des données entre machines aux différentes architectures
- Langage de description des données

Types

- Types primitifs proches du C
  - entiers, n'ombres à virgule flottante, caractères
- Structures de données
  - Tableaux, enregistrements, unions de types

NFS : sécurité ?

File access (véridique !)

- Client : Je voudrais accéder à ce fichier
- Serveur : OK, voilà les droits
- Client : J'ai vérifié, j'ai le droit d'y accéder
- Serveur : OK, voilà le fichier

Il suffit d'avoir les droits root sur la machine pour accéder à tout ...
Ou de contrôler le canal de communication !

NFS : identité contrôlée par le client

Fonction MNT du programme MOUNT

Requête

```
▼ Remote Procedure Call, Type:Call XID:0x14cf5b45
    XID: 0x14cf5b45 (349133637)
    Message Type: Call (0)
    RPC Version: 2
    Program: MOUNT (100005)
    Program Version: 3
    Procedure: MNT (1)
    [The reply to this request is in frame 28]
  ▼ Credentials
      Flavor: AUTH_UNIX (1)
      Length: 32
      Stamp: 0x0106295c
    ▶ Machine Name: batman
      UID: 0
      GID: 0
    ▶ Auxiliary GIDs (1) [0]
  ▶ Verifier
```

Réponse

```
▼ Remote Procedure Call, Type:Reply XID:0x14cf5b45
    XID: 0x14cf5b45 (349133637)
    Message Type: Reply (1)
    [Program: MOUNT (100005)]
    [Program Version: 3]
    [Procedure: MNT (1)]
    Reply State: accepted (0)
    [This is a reply to a request in frame 27]
    [Time from request: 0.000232438 seconds]
  ▶ Verifier
    Accept State: RPC executed successfully (0)
```

NFS : identité contrôlée par le client

Fonction READDIRPLUS du programme NFS

Requête

```
▼ Remote Procedure Call, Type:Call XID:0x8d561d3d
  ▶ Fragment header: Last fragment, 160 bytes
    XID: 0x8d561d3d (2371231037)
    Message Type: Call (0)
    RPC Version: 2
    Program: NFS (100003)
    Program Version: 3
    Procedure: READDIRPLUS (17)
    [The reply to this request is in frame 15]
  ▼ Credentials
      Flavor: AUTH_UNIX (1)
      Length: 64
      Stamp: 0x01062ad2
    ▶ Machine Name: batman
      UID: 1000
      GID: 1000
    ▶ Auxiliary GIDs (9) [24, 25, 27, 29, 30, 44, 46, 108, 1000]
  ▶ Verifier
```

Réponse

```
▼ Remote Procedure Call, Type:Reply XID:0x8d561d3d
  ▶ Fragment header: Last fragment, 488 bytes
    XID: 0x8d561d3d (2371231037)
    Message Type: Reply (1)
    [Program: NFS (100003)]
    [Program Version: 3]
    [Procedure: READDIRPLUS (17)]
    Reply State: accepted (0)
    [This is a reply to a request in frame 14]
    [Time from request: 0.000140663 seconds]
  ▶ Verifier
    Accept State: RPC executed successfully (0)
```

NFS : sécurité ?

NFSv4

Abandon du mount access par IP et du file access par auth. distante
Impose la mise en place d'un serveur Kerberos ... (on a le droit de rêver)

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

Attaque

- L'utilisateur `n7` sur `batman` monte un répertoire `home` de `robin`

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

Attaque

- L'utilisateur `n7` sur `batman` monte un répertoire `home` de `robin`
- Contient le `home` des utilisateurs `n7` et `test`

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

Attaque

- L'utilisateur `n7` sur `batman` monte un répertoire `home` de `robin`
- Contient le `home` des utilisateurs `n7` et `test`
- Accès non autorisé sur `/mountpoint/home/test`

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

Attaque

- L'utilisateur `n7` sur `batman` monte un répertoire `home` de `robin`
- Contient le `home` des utilisateurs `n7` et `test`
- Accès non autorisé sur `/mountpoint/home/test`
- Élévation de privilèges sur `batman`

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

Attaque

- L'utilisateur `n7` sur `batman` monte un répertoire `home` de `robin`
- Contient le `home` des utilisateurs `n7` et `test`
- Accès non autorisé sur `/mountpoint/home/test`
- Élévation de privilèges sur `batman`
- Création de l'utilisateur test et prise d'identité

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

Attaque

- L'utilisateur `n7` sur `batman` monte un répertoire `home` de `robin`
- Contient le `home` des utilisateurs `n7` et `test`
- Accès non autorisé sur `/mountpoint/home/test`
- Élévation de privilèges sur `batman`
- Création de l'utilisateur test et prise d'identité
- Accès autorisé sur `/mountpoint/home/test`

NFS : attaque de la gestion des droits UNIX

Hôte autorisé par IP

- Autorisé dans le fichier `/etc/exports`

Attaque

- L'utilisateur `n7` sur `batman` monte un répertoire `home` de `robin`
- Contient le `home` des utilisateurs `n7` et `test`
- Accès non autorisé sur `/mountpoint/home/test`
- Élévation de privilèges sur `batman`
- Création de l'utilisateur test et prise d'identité
- Accès autorisé sur `/mountpoint/home/test`

Analyse sécurité

- Pourquoi est-ce que cette attaque fonctionne ?

$\Rightarrow$ Conséquence ?

NFS : attaque de la gestion des droits UNIX

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`
- En outre, `robin` est autorisé

NFS : attaque de la gestion des droits UNIX

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`
- En outre, `robin` est autorisé

Attaque

- Usurpation locale de l'adresse IP de `robin` depuis `joker`

NFS : attaque de la gestion des droits UNIX

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`
- En outre, `robin` est autorisé

Attaque

- Usurpation locale de l'adresse IP de `robin` depuis `joker`
- Accès au service

NFS : attaque de la gestion des droits UNIX

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`
- En outre, `robin` est autorisé

Attaque

- Usurpation locale de l'adresse IP de `robin` depuis `joker`
- Accès au service

Analyse sécurité

- Pourquoi est-ce que cette attaque fonctionne ?

NFS : confidentialité par homme dans le milieu

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`

NFS : confidentialité par homme dans le milieu

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`

Attaque

- Homme dans le milieu local par `joker`

NFS : confidentialité par homme dans le milieu

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`

Attaque

- Homme dans le milieu local par `joker`
- Attente de l'accès au service par `batman` sur `robin`

NFS : confidentialité par homme dans le milieu

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`

Attaque

- Homme dans le milieu local par `joker`
- Attente de l'accès au service par `batman` sur `robin`
- Enregistrement du trafic réseau NFS

NFS : confidentialité par homme dans le milieu

Hôte non-autorisé par IP

- Non présent dans le fichier `/etc/exports`

Attaque

- Homme dans le milieu local par `joker`
- Attente de l'accès au service par `batman` sur `robin`
- Enregistrement du trafic réseau NFS

Analyse sécurité

- Pourquoi est-ce que cette attaque fonctionne ?

Un serveur graphique

Service

- Permet aux applications de dessiner sur un écran
- Permet aux applications d'interagir avec les périphériques d'entrée
- Serveur ?
  Services proposés aux applications

$\Rightarrow$ Développer des interfaces hommes machines

Triptyque classique

- Serveur graphique ou système de fenêtrage
- Client : gestionnaire de fenêtres
- Abstraction du paradigme : bibliothèque de *widgets*

Le serveur graphique X

Concept

- Tout est fenêtre
- Primitives de création / organisation / déplacement de fenêtres
- Primitives de dessin dans ces fenêtres
- Primitives d'abonnement aux évènements des périphériques

Optimisation

- `pixmap` : tampon permettant de pousser des matrices de pixels
  - Images
  - Accélération vidéo
- Primitives de copie de `pixmap` côté serveur

Le serveur graphique X

Principe

`DISPLAY = [hostname] : n`

Connexion des logiciels (clients) au serveur X

- par TCP/6000+n si hostname est précisé
- par un socket UNIX sinon (seulement en local)

Le serveur graphique X

Principe

`DISPLAY = [hostname] : n`

Connexion des logiciels (clients) au serveur X

- par TCP/6000+n si hostname est précisé
- par un socket UNIX sinon (seulement en local)

Configuration classique

- Flux TCP
- Accès distant contrôlé par IP (`xhost +ip`)

Démonstration : contrôle d'accès par IP

Configuration

- Machine virtuelle
- Activation de l'écoute TCP sur le port 6000
- Redirection de port et source NAT avec qemu

Démonstration : contrôle d'accès par IP

Configuration

- Machine virtuelle
- Activation de l'écoute TCP sur le port 6000
- Redirection de port et source NAT avec qemu

Manipulation

- Application hôte tente de se connecter au serveur graphique de la machine virtuelle
  - `$ export DISPLAY='locahost:0'`
- Ajout de l'accès pour l'adresse IP utilisée par le source NAT
  - `# xhost +10.0.2.0` ou
  - `# xhost +` pour désactivation
- Accès au service

Démonstration : `keylogger`

Configuration

- Machine virtuelle
- Activation de l'écoute TCP sur le port 6000
- Redirection de port et source NAT avec qemu

Démonstration : `keylogger`

Configuration

- Machine virtuelle
- Activation de l'écoute TCP sur le port 6000
- Redirection de port et source NAT avec qemu

Manipulation

- Application malveillante se connecte au serveur graphique de la machine virtuelle
  - `$ export DISPLAY='locahost:0'`
- Abonnement aux évènements "intéressants"
  - `$ xinput --test 'périphérique'`

Démonstration : écriture ”chez les copains”

Configuration

- Machine virtuelle
- Activation de l’écoute TCP sur le port 6000
- Redirection de port et source NAT avec qemu

Démonstration : écriture ”chez les copains”

Configuration

- Machine virtuelle
- Activation de l'écoute TCP sur le port 6000
- Redirection de port et source NAT avec qemu

Manipulation

- Application malveillante se connecte au serveur graphique de la machine virtuelle
  - `$ export DISPLAY='locahost:0'`
- Écriture dans la fenêtre victime
  - `$ lst/log-win.elf` Récupération de l'id de la victime
  - `$ lst/draw.elf` Exécution de l'attaque

Le serveur graphique X : sécurité ?

Analyse

Flux TCP

- Non chiffrés
- Non authentifiés

Conséquences

- N'importe qui peut injecter/lire des événements (clavier+souris+écran)
- X à énormément de privilèges sur le système ...

Le serveur graphique X : efforts

Configuration moderne

Utilisation de cookies d'authentification (`xauth`)
... Utilisant le protocol MIT-MAGIC-COOKIE-1 (envoi en clair du mdp)

Verrouillage

1. Debian/Ubuntu : `-nolisten tcp` (voir `/etc/X11/xinit/xserverrc`)
   Impossible d'utiliser X pour des applications à distance
2. Utilisation d'un autre protocole pour xauth (Kerberos ?)

Plan

En amont

Initialisation

Utilisation d'une authentification par mot de passe / clé

Ne doit pas se baser sur l'incapacité technique à faire quelque chose (e.g. on sait qu'il suffit de modifier l'adresse IP mais l'OS nous en empêche)

Messages échangés

Pour ceux dont on a besoin d'être sûrs de l'origine MAC
Si besoin de confidentialité chiffrement authentifié

Il faut un secret commun ! (ev. fourni par l'authentification)

En aval

Encapsulation dans un tunnel chiffré

Authentification + échange de clés
Création d'un tunnel dans lequel on encapsule:

- Une connexion TCP (i.e. pour un port source/dest IP dest donnés)
- Tous les datagrammes IP à destination une plage IP donnée
- Toutes les trames correspondant à une liste d'adresses destination

Dans ce tunnel les données sont chiffrés et authentifiées

Attention la notion de tunnel sécurisé existe pour toutes les couches
$\rightarrow$ MACSec, IPSec, TLS, protocole SSH, etc.

Plan

Fin !

Prochain cours

Exam ?