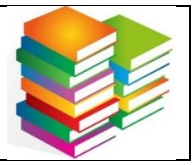


Les filtres d'affichage Wireshark



Une capture peut contenir des centaines de paquets. Un filtre d'affichage permet de ne montrer que ceux qui correspondent à un critère (protocole, adresse IP, port, type de message, ..)

Utiliser un filtre

1. Ouvrir la capture dans Wireshark
2. Saisir une expression dans la barre de filtre d'affichage, au-dessus de la liste des paquets
3. Appuyer sur Entrée pour l'appliquer
4. Effacer le filtre pour revenir à la capture complète

Un fond vert indique une syntaxe valide.
Un fond rouge indique une erreur.

Attention, les filtres d'affichage et les filtres de capture utilisent des syntaxes différentes. Cette filtre concerne uniquement les filtres d'affichage.

Filtre par protocole

Saisir simplement le nom du protocole en minuscule

Filtre	Paquets affichés
`arp`	Recherche et annonce d'adresses MAC en IPv4
`dns`	Requêtes et réponses DNS
`icmp`	Messages ICMP, notamment les échanges de ping IPv4
`tcp`	Trafic utilisant TCP
`udp`	Trafic utilisant UDP
`http`	Messages décodés comme HTTP
`tls`	Messages décodés comme TLS, utilisé notamment pour HTTPS

Exemple : dns permet de rechercher les noms demandés par les postes et les réponses reçues.

Filtrer par adresse IP

Besoin	Filtre
Voir les échanges impliquant un poste	`ip.addr == 192.168.64.123`
Voir les paquets envoyés par ce poste	`ip.src == 192.168.64.123`
Voir les paquets destinés à ce poste	`ip.dst == 192.168.64.123`
Voir le trafic impliquant un sous-réseau	`ip.addr == 192.168.64.0/24`
Masquer les échanges impliquant ce poste	`!(ip.addr == 192.168.64.123)`

Mémo :

src = source, celui qui envoie
dst = destination, celui qui reçoit
addr = source ou destination

A. BÉGER

Filtrer par port

Un port aide à identifier le service sollicité.

Besoin	Filtre
TCP avec un port source ou destination égale à 80	`tcp.port == 80`
TCP vers le port 80 du serveur	`tcp.dstport == 80`
TCP provenant du port 80 du serveur	`tcp.srcport == 80`
UDP avec un port source ou destination égal à 53	`udp.port == 53`

Combiner plusieurs critères

Opérateur	Signification	Exemple
`==`	Egal à	`tcp.port == 80`
`&&`	ET : les deux conditions sont vraies	`ip.addr == 192.168.64.123 && http`
`||`	OU : au moins une condition doit être vraie	`arp || dns`
`!`	NON : exclut une condition	`!arp`
`>` ou `<`	Supérieur ou inférieur à	`frame.len > 1000`

Exemples

Afficher uniquement les échanges entre deux équipements dans les deux sens :

```
ip.addr == 192.168.64.123 && ip.addr==192.168.64.147
```

Afficher les paquets envoys par un poste vers un serveur :

```
ip.src == 192.168.64.123 && ip.dst == 192.168.64.147
```

Afficher le trafic http ou DNS impliquant un poste :

```
ip.addr == 192.168.64.123 && (http || dns)
```

Examiner les échanges HTTP

Ces filtres sont utiles lorsque Wireshark peut lire et décoder les messages HTTP.

Besoin	Filtre
Toutes les requêtes HTTP	`http.request`
Toutes les réponses HTTP	`http.response`
Les demandes GET	`http.request.method == "GET"`
Les envois POST	`http.request.method == "POST"`
Les demandes vers /login	`http.request.uri == "/login"`
Les adresses demandées contenant « exports »	`http.request.uri contains "exports"`
Les codes réponses (200 par exemple)	`http.response.code == 200`

Exemple : repérer les formulaires envoyés à la page de connexion

```
http.request.method == "POST" && http.request.uri == "/login"
```

Ce filtre affiche les requêtes. Pour examiner leurs réponses, suivre le flux TCP correspondant.

Isoler une connexion ou une période

Besoin	Filtre
Demandes initiales de connexion TCP	`tcp.flags.syn == 1 && tcp.flags.ack == 0`
Réinitialisations de connexion TCP	`tcp.flags.reset == 1`
Une conversation TCP précise	`tcp.stream == 0`
Une trame précise	`frame.number == 100`
Les trames entre les numéros 100 et 150 inclus	`frame.number >= 100 && frame.number <= 150`
Les paquets entre 60 et 100 secondes depuis le début	`frame.time_relative >= 60 && frame.time_relative <= 100`

Le numéro dans `tcp.stream == 0` est un exemple, chaque conversation possède son propre numéro dans la capture.

Pour isoler facilement une conversation :

1. Sélectionner un paquet
2. Faire un clic droit > Suivre > Flux TCP
3. Lire les échanges entre le client et le serveur
