Wireshark – это одна из наиболее известных и лучших в мире программа-анализатор (сниффер) для захвата и декодирования сетевого трафика. Она предоставляет возможность декодировать более 500 различных протоколов сетей передачи данных и телекоммуникационных протоколов, включая протоколы сотовой связи. Она является де-факто (и часто де-юре) стандартом во многих отраслях промышленности и образовательных учреждениях во всем мире. Многие производители коммерческих продуктов используют его в своих решениях как декодировщик.

Любой анализатор протоколов должен иметь возможность не только захватить трафик, но и помочь эффективно его проанализировать. В сетях передачи данных на скоростях 1 Гбит/сек и выше буфер захвата трафика заполняется мгновенно и на выходе получается достаточно большой массив данных. Для анализа, этот массив данных можно отфильтровать по разным параметрам, что в Wireshark реализовано следующим функционалом:

- Цветовая кодировка ошибочных пакетов — можно настроить под себя. Пакеты, которые несут в себе ошибку, будут выделены в буфере специальным цветом.
- Фильтр через строку фильтрации. Вы имеете большой опыт в работе с Wireshark и протоколами и можете ввести фильтр самостоятельно. Большой выбор фильтров можно найти здесь.
- Выделение любой области в пакете, правый клик мыши и «Применить как фильтр».

Wireshark - фильтрация по протоколу

Достаточно в строке фильтра ввести название протокола и нажать ввод. На экране останутся пакеты, которые относятся к искомому протоколу. Таким образом, фильтр выглядит:

```
http
```

Результат применения фильтра:

Если буфер захвата необходимо отфильтровать по нескольким протоколам, то необходимо перечислить все желаемые протоколы и разделить их знаком ||. Например:

```
arp || http || icmp
```

Результат применения фильтра:

Wireshark - фильтрация по IP адресу и MAC адресу

В зависимости от направления трафика фильтр будет немного отличаться. Например, мы хотим отфильтровать по IP адресу отправителя 10.0.10.163:

```
ip.src==10.0.10.163
```

Результат применения фильтра:

По получателю фильтр будет выглядеть ip.dst == x.x.x.x, а если хотим увидеть пакеты в независимости от направления трафика, то достаточно ввести:

```
ip.addr==50.116.24.50
```

В случае если нам необходимо исключить какой то адрес из поля отбора, то необходимо добавить != . Пример:

```
ip.src!=80.68.246.17
```

Результат применения фильтра:

Если мы анализируем трафик внутри локальной сети и знаем MAC адрес пользователя, то можно указать в качестве фильтра Wireshark его MAC адрес, например:

```
eth.addr == AA:BB:CC:DD:EE:FF
```

Wireshark - фильтрация по номеру порта

При анализе трафика мы можем настроить фильтр по номеру порта, по которому осуществляет передачу трафика тот или иной протокол. Номера всех зарегистрированных портов можно узнать здесь. Пример:

```
ftp.port==21
```

Так же как и с адресами IP и MAC мы можем отдельно фильтровать по портам получения или отправления **tcp.srcport** и **tcp.dstport.** Кроме указания номеров портов Wireshark дает отличную возможность отфильтровать буфер по флагам в TCP протоколе. Например, если мы хотим увидеть TCP пакеты с флагом SYN (установление соединения между устройствами), то вводим в строке поиска:

```
tcp.flags.syn
```

Результат применения фильтра:

Популярные фильтры

В таблице ниже приведены наиболее популярные фильтры для отображения содержимого буфера захвата:

Фильтр для отображения	Описание	Пример написания
eth.addr	MAC адрес отправителя или получателя	eth.addr == 00:1a:6b:ce:fc:bb
eth.src	MAC-адрес оправителя	eth.src == 00:1a:6b:ce:fc:bb
eth.dst	MAC-адрес получателя	eth.dst == 00:1a:6b:ce:fc:bb
arp.dst.hw_mac	Протокол ARP – MAC адрес получателя	arp.dst.hw_mac == 00:1a:6b:ce:fc:bb
arp.dst.proto_ipv4	Протокол ARP – IP адрес версии 4 получателя	arp.dst.proto_ipv4 == 10.10.10.10
arp.src.hw_mac	Протокол ARP – MAC адрес отправителя	arp.src.hw_mac == 00:1a:6b:ce:fc:bb
arp.src.proto_ipv4	Протокол ARP – IP адрес версии 4 отправителя	arp.src.proto_ipv4 == 10.10.10.10
vlan.id	Идентификатор VLAN	vlan.id == 16
ip.addr	IP адрес версии 4 получателя или отправителя	ip.addr == 10.10.10.10
ip.dst	IP адрес версии 4 получателя	ip.addr == 10.10.10.10
ip.src	IP адрес версии 4 отправителя	ip.src == 10.10.10.10
ip.proto	IP protocol (decimal)	ip.proto == 1
ipv6.addr	IP адрес версии 6 получателя или отправителя	ipv6.addr == 2001::5
ipv6.src	IP адрес версии 6 отправителя	ipv6.addr == 2001::5
ipv6.dst	IP адрес версии 6 получателя	ipv6.dst == 2001::5
tcp.port	TCP порт получателя или отправителя	tcp.port == 20

tcp.dstport	TCP порт получателя	tcp.dstport == 80
tcp.srcport	TCP порт отправителя	tcp.srcport == 60234
udp.port	UDP порт получателя или отправителя	udp.port == 513
udp.dstport	UDP порт получателя	udp.dstport == 513
udp.srcport	UDP порт отправителя	udp.srcport == 40000
vtp.vlan_info.vlan_name	Имя VLAN	vtp.vlan_info.vlan_name == TEST
bgp.originator_id	Идентификатор BGP (Адрес IPv4)	bgp.originator_id == 192.168.10.15
bgp.next_hop	Следующий хоп BGP (Адрес IPv4)	bgp.next_hop == 192.168.10.15
rip.ip	RIP IPv4 address	rip.ip == 200.0.2.0
ospf.advrouter	Идентификатор маршрутизатора по протоколу OSPF	ospf.advrouter == 192.168.170.8
eigrp.as	Номер автономной системы EIGRP	eigrp.as == 100
hsrp.virt_ip	Виртуальный IP адрес по протоколу HSRP	hsrp.virt_ip == 192.168.23.250
vrrp.ip_addr	Виртуальный IP адрес по протоколу VRRP	vrrp.ip_addr == 192.168.23.250
wlan.addr	MAC адрес отправителя или получателя Wi-Fi	wlan.addr == 00:1a:6b:ce:fc:bb
wlan.sa	MAC-адрес оправителя Wi-Fi	wlan.sa == 00:1a:6b:ce:fc:bb
wlan.da	MAC-адрес получателя Wi-Fi	wlan.da == 00:1a:6b:ce:fc:bb