Netdata, alertes i nmap

Activitat: Crea una alerta per detectar un escanig de més de 100 ports.

Índex

- 1. Primer investiga què és un TCP Reset. Fes-ne una petita explicació i digue's en quins casos s'utilitza.
- 2. El Netdata té alarmes creades per detectar TCP resets. Estan definides al fitxer *tcp_resets.conf*. Prova de fer nmaps a la teva màquina on hi tens el Netdata. Adjunta una captura de l'alerta que has rebut.
- 3. Després d'estudiar i entendre com funcionen les alarmes creades per Netdata sobre TCP resets, crea una alerta al mateix fitxer *tcp_resets.conf* que comenci amb el teu nom (per exemple elteunom_mes_de_100_TCP_Resets) i que t'avisi quan la teva màquina enviï més de 100 tcp resets seguits. Adjunta una captura de l'alerta que acabes de fer.
- 4. Adjunta una captura de la teva alerta que has rebut per correu electrònic.
- 5. Completa el teu exercici utilitzant el Wireshark amb el filtre adequat per obtenir la IP de la màquina que t'està fent l'nmap. Adjunta també una captura amb el filtre i la IP de l'atacant.

1. Primer investiga què és un TCP Reset. Fes-ne una petita explicació i digue's en quins casos s'utilitza.

Un TCP Reset és un paquet, també anomenat segment, que s'utilitza en el protocol TCP per reiniciar la connexió.

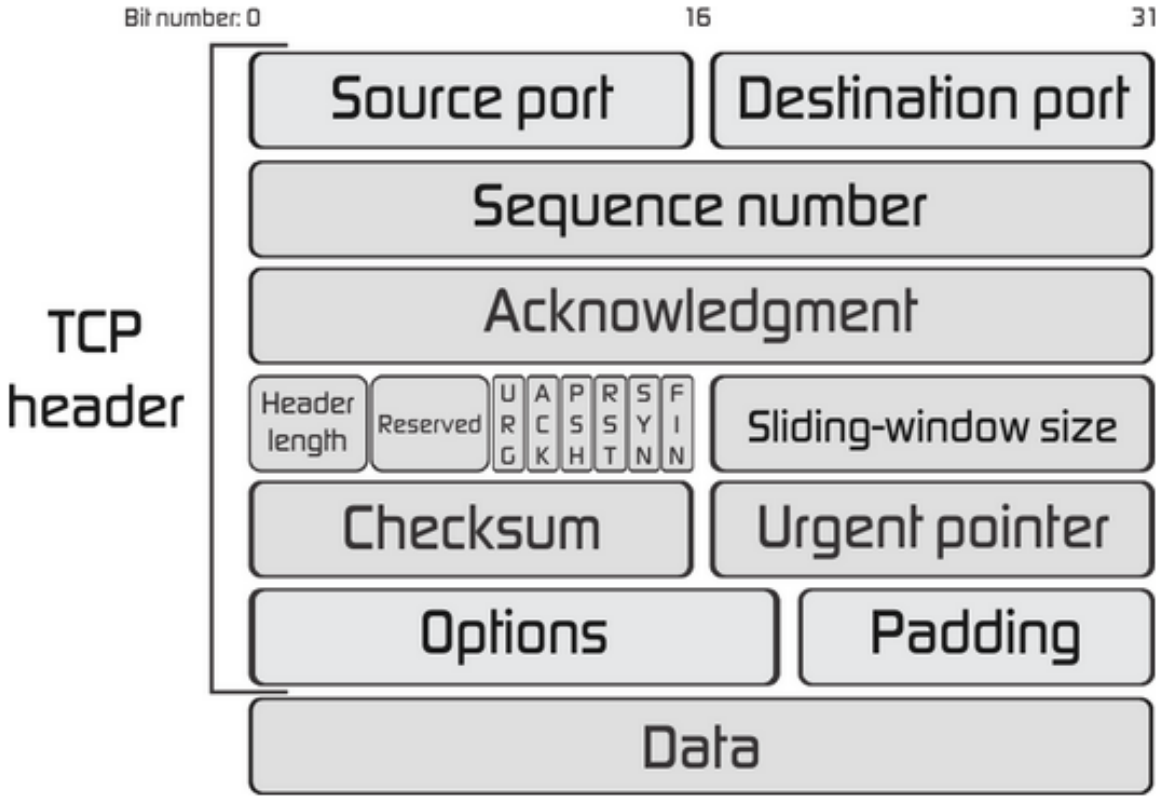


Figure 1: "Estructura d'un segment TCP"

Imatge extreta de: https://fiberbit.com.tw/tcp-transmission-control-protocol-segments-and-fields/

Dins del segment trobem a la capçalera el camp *flags*, dividit en 6 bits. Aquests bits són els que ens trobem a la imatge:

- URG
- ACK
- PSH
- RST
- SYN
- FIN

Quan el bit RST té valor 1, el paquet està demanant que cal reiniciar la connexió. S'utilitza quan es vol rebutjar un intent de connexió.

Hi ha un tipus de ciberatac on s'utilitza aquesta funcionalitat. Aquest atac és el que fan servir països com China o Iran per interferir i bloquejar les connexions per tal de censurar internet.

Quan es troben molts TCP Resets pot indicar que s'està realitzant un escaneig de ports.

2. El Netdata té alarmes creades per detectar TCP resets. Estan definides al fitxer *tcp_resets.conf*. Prova de fer nmaps a la teva màquina on hi tens el Netdata. Adjunta una captura de l'alerta que has rebut.

Aquestes són les configuracions de les alertes del Netdata:

```
bash-5.0# pwd
/usr/lib/netdata/conf.d/health.d
bash-5.0# ls
adaptec_raid.conf        exporting.conf           memcached.conf           systemdunits.conf
anomalies.conf           fping.conf               memory.conf              tcp_conn.conf
apcupsd.conf             fronius.conf             mysql.conf               tcp_listen.conf
backend.conf             gearman.conf             net.conf                 tcp_mem.conf
bcache.conf              geth.conf                netfilter.conf           tcp_orphans.conf
beanstalkd.conf          go.d.plugin.conf         pihole.conf              tcp_resets.conf
bind_rndc.conf           haproxy.conf             portcheck.conf           timex.conf
boinc.conf               hdfs.conf                processes.conf           udp_errors.conf
btrfs.conf               httpcheck.conf           python.d.plugin.conf     unbound.conf
ceph.conf                ioping.conf              qos.conf                 vcsa.conf
cgroups.conf             ipc.conf                 ram.conf                 vernemq.conf
cockroachdb.conf         ipfs.conf                redis.conf               vsphere.conf
cpu.conf                 ipmi.conf                retroshare.conf          web_log.conf
dbengine.conf            isc_dhcpd.conf           riakkv.conf              whoisquery.conf
disks.conf               kubelet.conf             scaleio.conf             wmi.conf
dns_query.conf           linux_power_supply.conf  softnet.conf             x509check.conf
dnsmasq_dhcp.conf        load.conf                stiebeleltron.conf       zfs.conf
dockerd.conf             mdstat.conf              swap.conf
entropy.conf             megacli.conf             synchronization.conf
```

Figure 2: "Configuracions d'alertes"

Contingut del fitxer *tcp_resets.conf*:

```
# you can disable an alarm notification by setting the 'to' line to: silent

#   ----------------------------------------------------------------------

# tcp resets this host sends

     alarm: 1m_ipv4_tcp_resets_sent
        on: ipv4.tcphandshake
     class: Errors
      type: System
 component: Network
        os: linux
     hosts: *
    lookup: average -1m at -10s unaligned absolute of OutRsts
     units: tcp resets/s
     every: 10s
      info: average number of sent TCP RESETS over the last minute

     alarm: 10s_ipv4_tcp_resets_sent
        on: ipv4.tcphandshake
     class: Errors
      type: System
 component: Network
        os: linux
```

```
    hosts: *
   lookup: average -10s unaligned absolute of OutRsts
    units: tcp resets/s
    every: 10s
     warn: $this > (((($1m_ipv4_tcp_resets_sent < 5)?(5):(
           $1m_ipv4_tcp_resets_sent)) * (($status >= $WARNING)  ? (1) :
           (20)))
    delay: up 20s down 60m multiplier 1.2 max 2h
  options: no-clear-notification
     info: average number of sent TCP RESETS over the last 10 seconds.
          \
           This can indicate a port scan, \
           or that a service running on this host has crashed. \
           Netdata will not send a clear notification for this alarm.
       to: sysadmin

#    ------------------------------------------------------------------------------

# tcp resets this host receives

    alarm: 1m_ipv4_tcp_resets_received
       on: ipv4.tcphandshake
    class: Errors
     type: System
component: Network
       os: linux freebsd
    hosts: *
   lookup: average -1m at -10s unaligned absolute of AttemptFails
    units: tcp resets/s
    every: 10s
     info: average number of received TCP RESETS over the last minute

    alarm: 10s_ipv4_tcp_resets_received
       on: ipv4.tcphandshake
    class: Errors
     type: System
component: Network
       os: linux freebsd
    hosts: *
   lookup: average -10s unaligned absolute of AttemptFails
    units: tcp resets/s
    every: 10s
     warn: $this > (((($1m_ipv4_tcp_resets_received < 5)?(5):(
           $1m_ipv4_tcp_resets_received)) * (($status >= $WARNING)  ? (1)
            : (10)))
    delay: up 20s down 60m multiplier 1.2 max 2h
  options: no-clear-notification
     info: average number of received TCP RESETS over the last 10
           seconds. \
           This can be an indication that a service this host needs
              has crashed. \
           Netdata will not send a clear notification for this alarm.
       to: sysadmin
```

> He provat vàries vegades a fer aquesta prova amb la instal·lació feta amb Docker, amb diferents configuracions pel contenidor i en diferents màquines.
>
> No he estat capaç de poder fer la prova amb èxit.
>
> Finalment he fet la prova amb la instal·lació comuna, sense fer servir Docker i m'ha funcionat la prova. No investigaré el motiu pel qual no funcionava amb Docker i directament faré servir el Netdata que he instal·lat amb la comanda:

```
bash <(curl -Ss https://my-netdata.io/kickstart.sh) --claim-token
    iCheH8mEpkkvtXbVAXkimEXdqUU-
    TwFmuPSbxrflzffz762tEh6ihTuXymdhQy7qurIUZpF0_Ek_Ti6nE5KYW4H-
    xqdITU5EaQCJ-iXslORqilzIzci_ICbXhQCVMa-CMd_EWIk --claim-rooms
    80859831-206c-4ade-8381-564f89751c9c --claim-url https://app.
    netdata.cloud
```

Com a sudo:

```
# Faig un bucle amb nmap -sS (sondeig TCP SYN)
# Primer ho he provat des de la mateixa màquina amb localhost
while true; do nmap -sS localhost; done

# Després ho he provat des d'una altra màquina.
while true; do nmap -sS 192.168.1.19; done
```

Moment en que salta l'alerta:

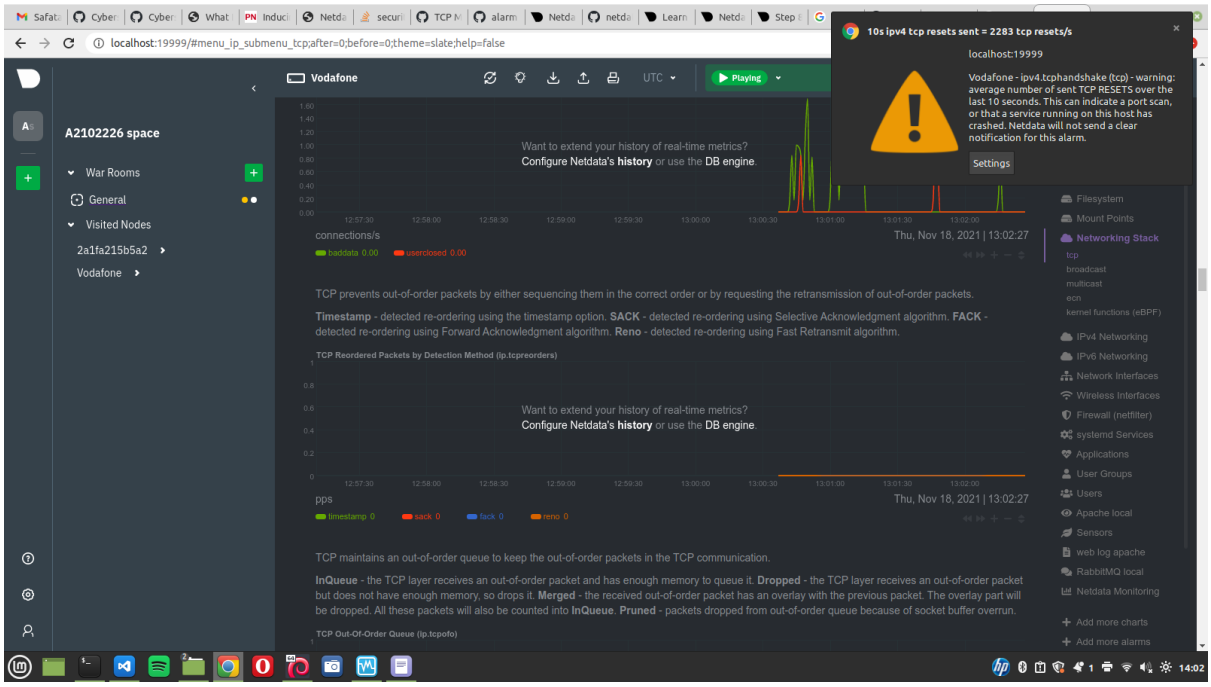


Figure 3: "Moment en que salta l'alerta"

A les següents captures es poden veure dos moments amb incidències, fent que salti l'alerta dues vegades, per l'intent fet en la mateixa màquina i en l'altra màquina en local.

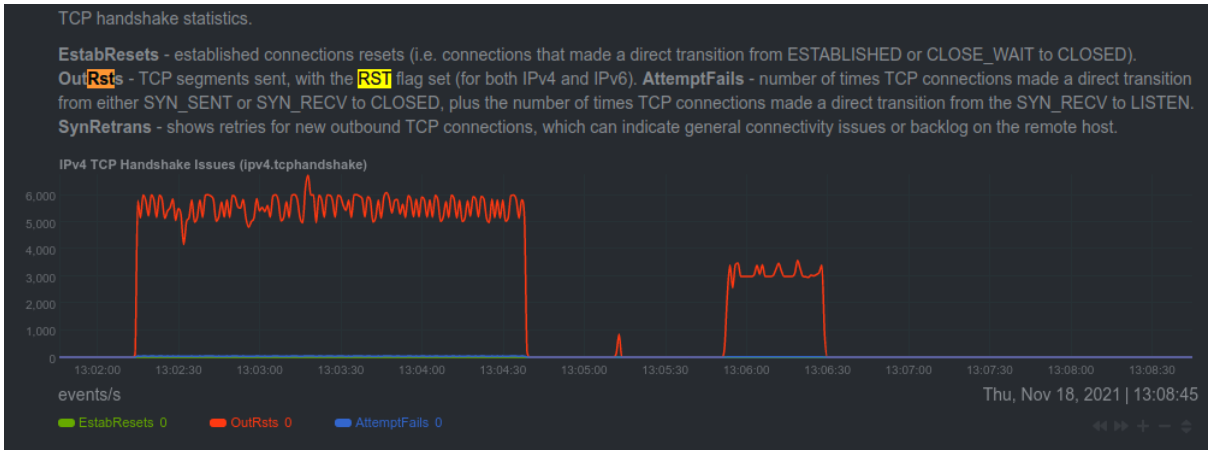
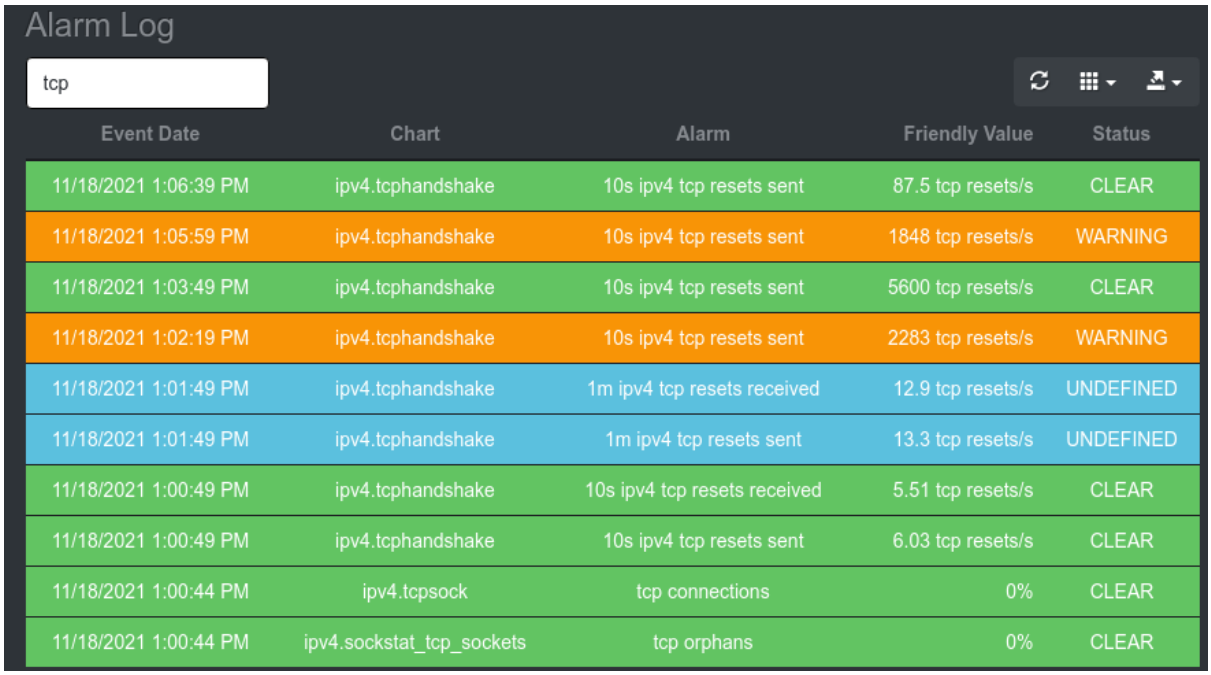


Figure 4: "Gràfic incidències TCP"



Event Date	Chart	Alarm	Friendly Value	Status
11/18/2021 1:06:39 PM	ipv4.tcphandshake	10s ipv4 tcp resets sent	87.5 tcp resets/s	CLEAR
11/18/2021 1:05:59 PM	ipv4.tcphandshake	10s ipv4 tcp resets sent	1848 tcp resets/s	WARNING
11/18/2021 1:03:49 PM	ipv4.tcphandshake	10s ipv4 tcp resets sent	5600 tcp resets/s	CLEAR
11/18/2021 1:02:19 PM	ipv4.tcphandshake	10s ipv4 tcp resets sent	2283 tcp resets/s	WARNING
11/18/2021 1:01:49 PM	ipv4.tcphandshake	1m ipv4 tcp resets received	12.9 tcp resets/s	UNDEFINED
11/18/2021 1:01:49 PM	ipv4.tcphandshake	1m ipv4 tcp resets sent	13.3 tcp resets/s	UNDEFINED
11/18/2021 1:00:49 PM	ipv4.tcphandshake	10s ipv4 tcp resets received	5.51 tcp resets/s	CLEAR
11/18/2021 1:00:49 PM	ipv4.tcphandshake	10s ipv4 tcp resets sent	6.03 tcp resets/s	CLEAR
11/18/2021 1:00:44 PM	ipv4.tcpsock	tcp connections	0%	CLEAR
11/18/2021 1:00:44 PM	ipv4.sockstat_tcp_sockets	tcp orphans	0%	CLEAR

Figure 5: "Log alertes"

Aquest és el correu rebut amb l'alerta:

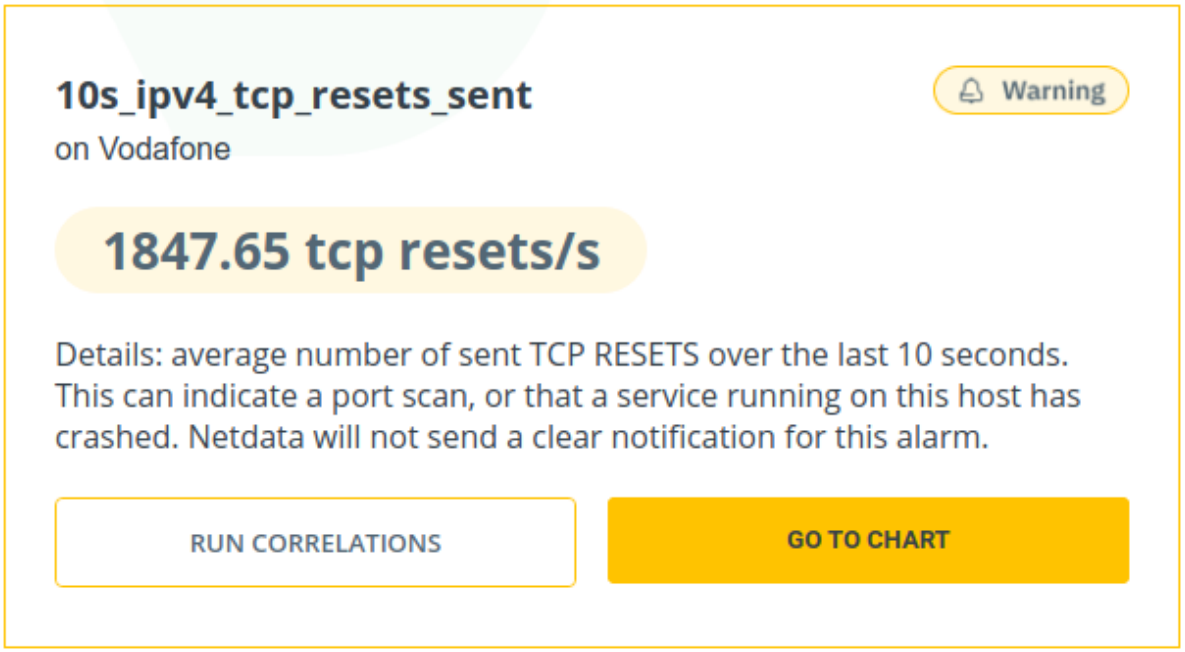


Figure 6: "Correu amb l'alerta"

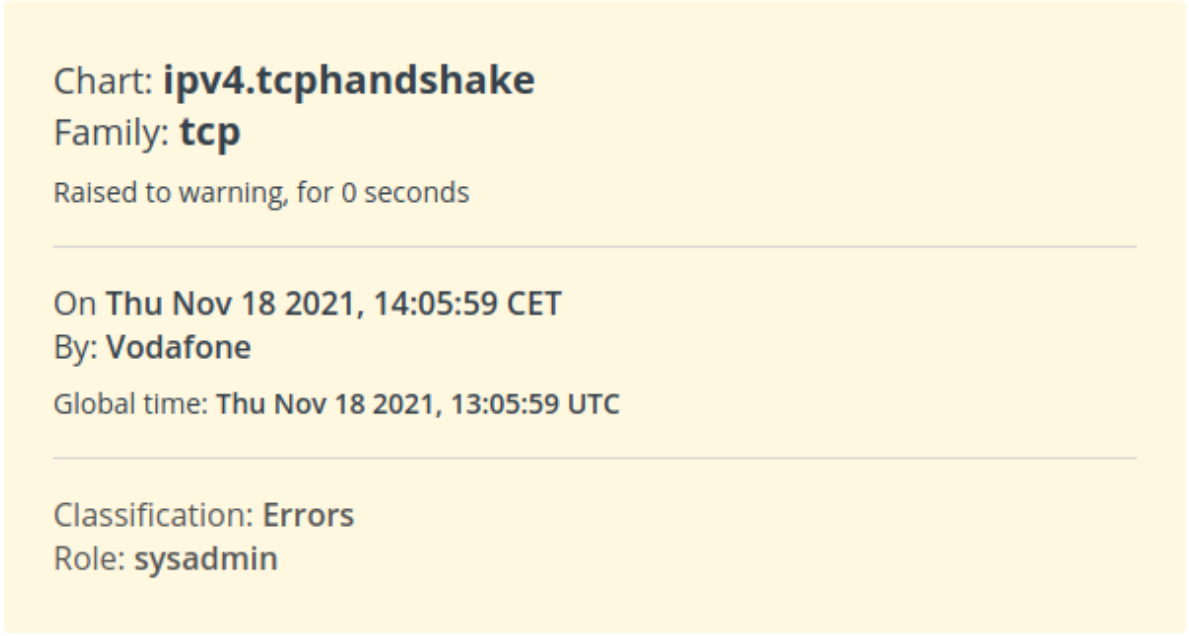


Figure 7: "Correu amb l'alerta"

3. Després d'estudiar i entendre com funcionen les alarmes creades per Netdata sobre TCP resets, crea una alerta al mateix fitxer *tcp_resets.conf* que comenci amb el teu nom (per exemple elteunom_mes_de_100_TCP_Resets) i que t'avisi quan la teva màquina enviï més de 100 tcp resets seguits. Adjunta una captura de l'alerta que acabes de fer.

```
1  # Editem el fitxer per afegir alerta.
```

```
sudo vi /usr/lib/netdata/conf.d/health.d/tcp_resets.conf

# Toni Peraira Alert
# Cada 100 TCP Resets en 5 segons, salta l'alerta amb estat WARNING.
# Si arriben a 1000, salta l'alerta amb estat crític.
    alarm: toni_peraira_mes_de_100_TCP_Resets
       on: ipv4.tcphandshake
    class: Errors
     type: System
component: Network
       os: linux
    hosts: *
   lookup: sum -5s unaligned absolute of OutRsts
    units: tcp resets/s
     warn: $this > 100
     crit: $this > 1000
     info: more than 100 TCP resets have been sent in a row, \
            over the last 5 seconds. \
            This can indicate a port scan, \
            or that a service running on this host has crashed.
       to: sysadmin
```

```
# Reiniciem
sudo netdatacli reload-health
```

Informació de l'alerta en Netdata:

Figure 8: "Informació de l'alerta en Netdata"

4. Adjunta una captura de la teva alerta que has rebut per correu electrònic.

Logs de l'alerta:

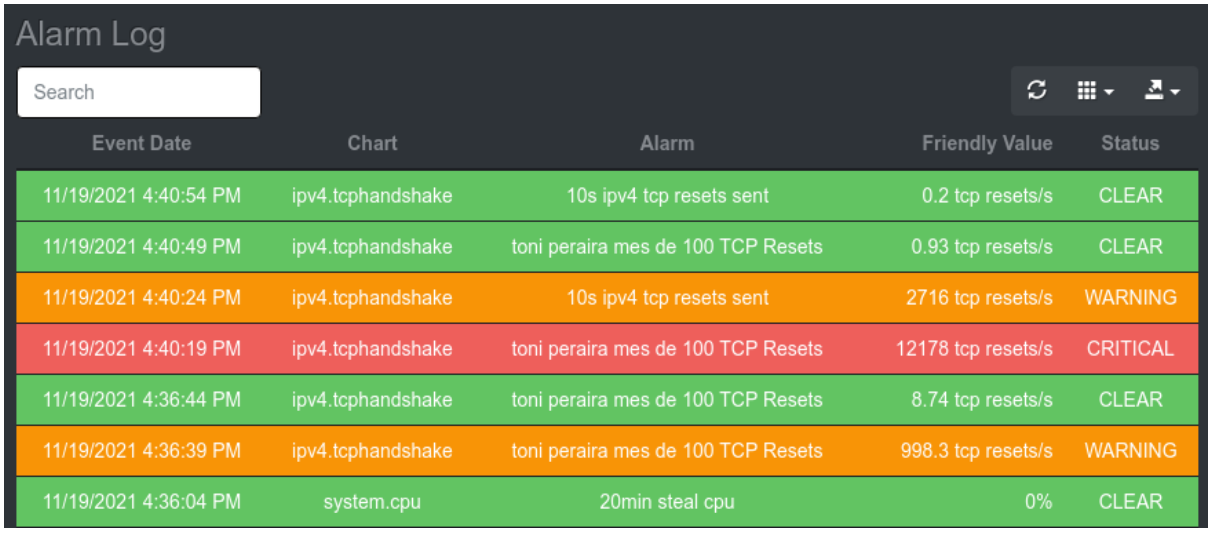


Figure 9: “Logs de l’alerta”

Moment en que salta una notificació d’alerta crítica:

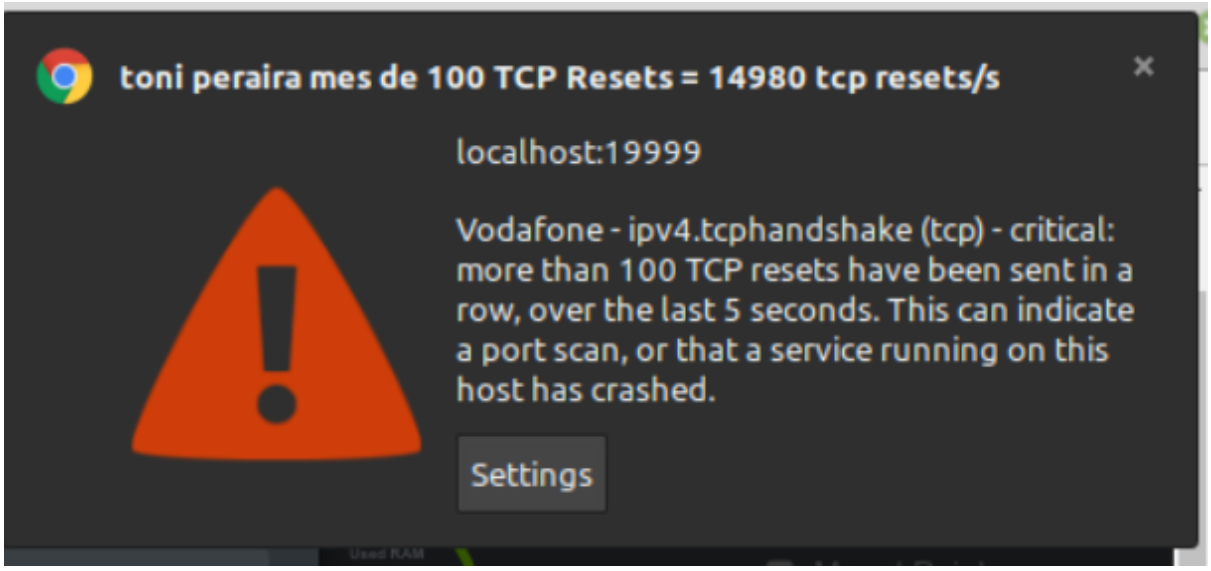


Figure 10: “Moment en que salta una notificació d’alerta crítica”

Moment en que salta una notificació que tot ha tornat a la normalitat:

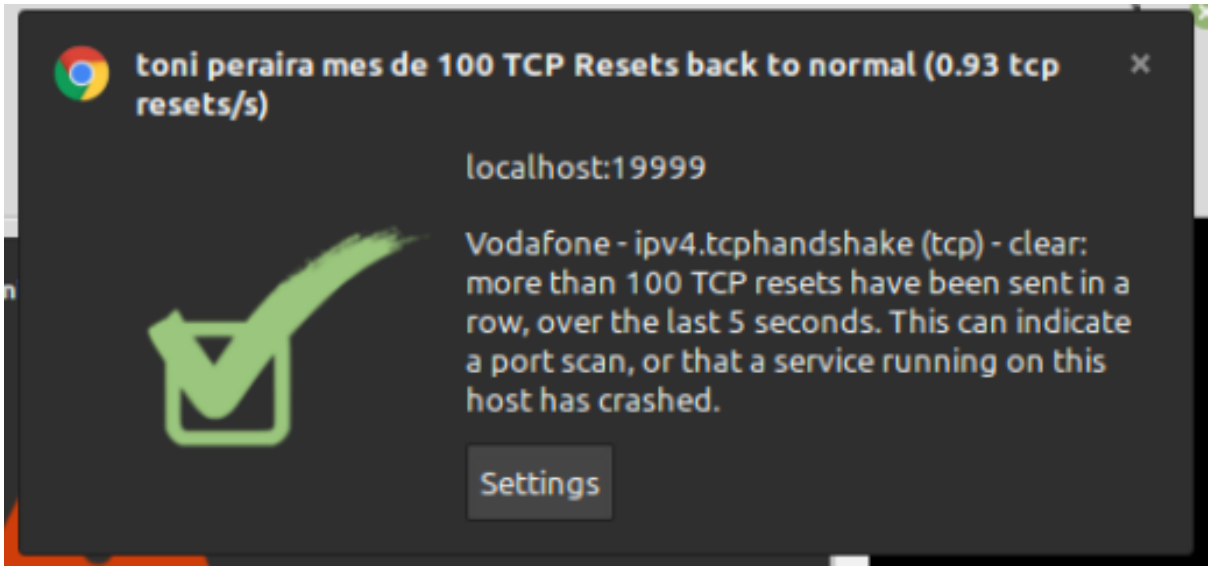


Figure 11: “Moment en que salta una notificació que tot ha tornat a la normalitat”

Correus rebuts:

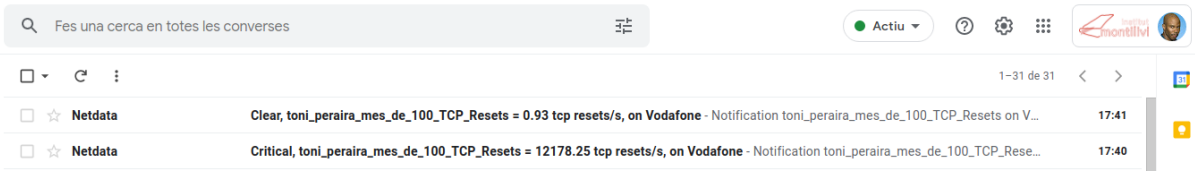


Figure 12: “Correus rebuts”

Correu amb l’alerta crítica:

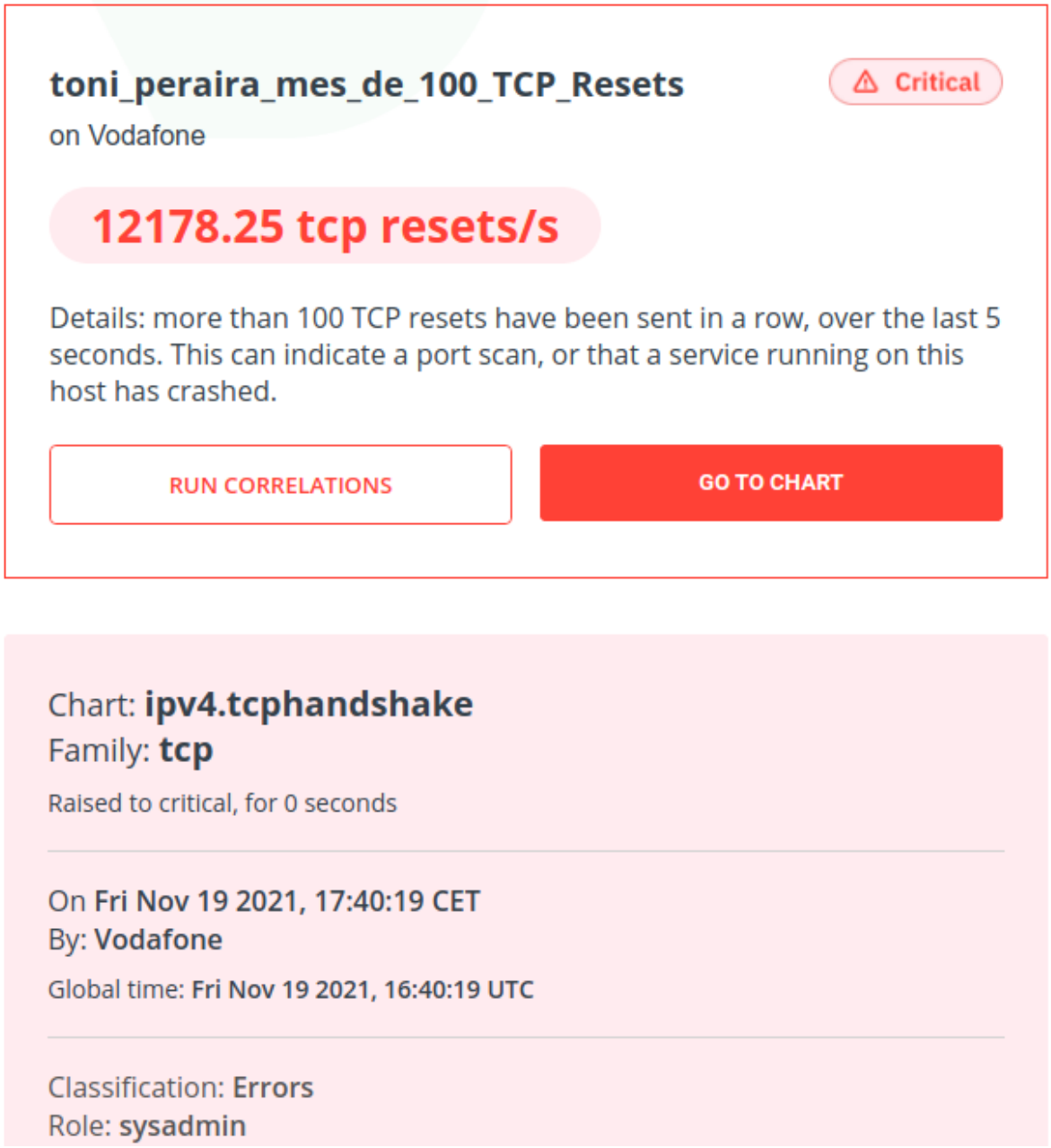


Figure 13: "Correu amb l'alerta crítica"

5. Completa el teu exercici utilitzant el Wireshark amb el filtre adequat per obtenir la IP de la màquina que t'està fent l'nmap. Adjunta també una captura amb el filtre i la IP de l'atacant.

Hi ha moltes formes de filtrar el Wireshark per detectar un possible escaneig de ports.

He fet servir el filtre:

```
tcp && tcp.flags.reset == 1 && tcp.window_size <=1024 && ip.src ==
    192.168.2.119
```

On:

> - tcp: Filtrem pel protocol TCP.
> - tcp.flags.reset == 1: Com estem fent exercicis sobre TCP Resets, filtrem pel *flag reset*, el qual hem descrit al principi.
> - tcp.windows_size <= 1024: L'escaneig de ports amb Nmap o altres eines no envien gaires dades, únicament envia les dades essencials per comprovar que existeix el port. Amb aquest filtre, filtrem paquets de poca grandària.
>
> ip.src == 192.168.2.119: És la IP de la màquina que rep l'escaneig de ports.

Aquí es mostren alguns filtres d'utilitat:

Detection of network port scanning

This section contains Wireshark filters useful for identifying various network port scans, port sweeps etc.

Here's the summary table with more details further down below:

Technique	Wireshark Filter	Command / Tool
TCP SYN scan	tcp.flags.syn==1 and tcp.flags.ack==0 and tcp.window_size<=1024	nmap -sS <target>
TCP Connect() scan	tcp.flags.syn==1 and tcp.flags.ack==0 and tcp.window_size>1024	nmap -sT <target>
TCP Null scan	tcp.flags==0	nmap -sN <target>
TCP FIN scan	tcp.flags==0x001	nmap -sF <target>
TCP Xmass scan	tcp.flags.fin==1 && tcp.flags.push==1 && tcp.flags.urg==1	nmap -sX <target>
UDP port scan	icmp.type==3 and icmp.code==3	nmap -sU <target>

Figure 14: "Wireshark filtres d'utilitat"

Imatge extreta de: https://www.infosecmatter.com/detecting-network-attacks-with-wireshark/

Aquí tenim el filtre preparat, però no hem executat l'escaneig de ports encara:

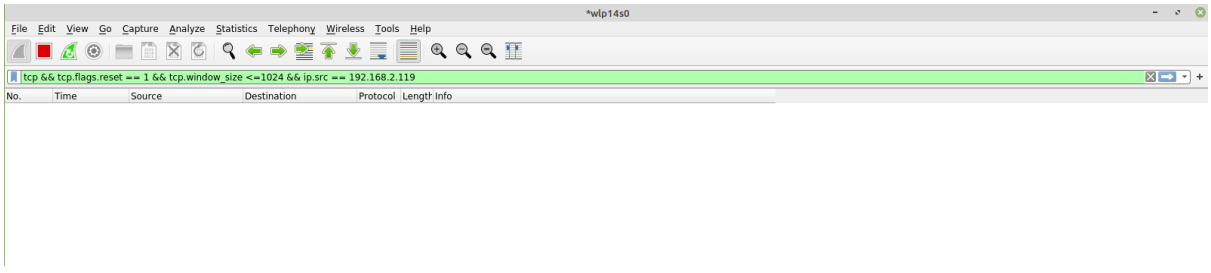


Figure 15: "Wireshark filtres d'utilitat"

En el moment en què executem l'escaneig de ports ja comencem a veure tots aquells TCP Resets que han succeït després de l'intent d'escaneig per part de la IP **192.168.2.196**, que és la meva màquina atacant:

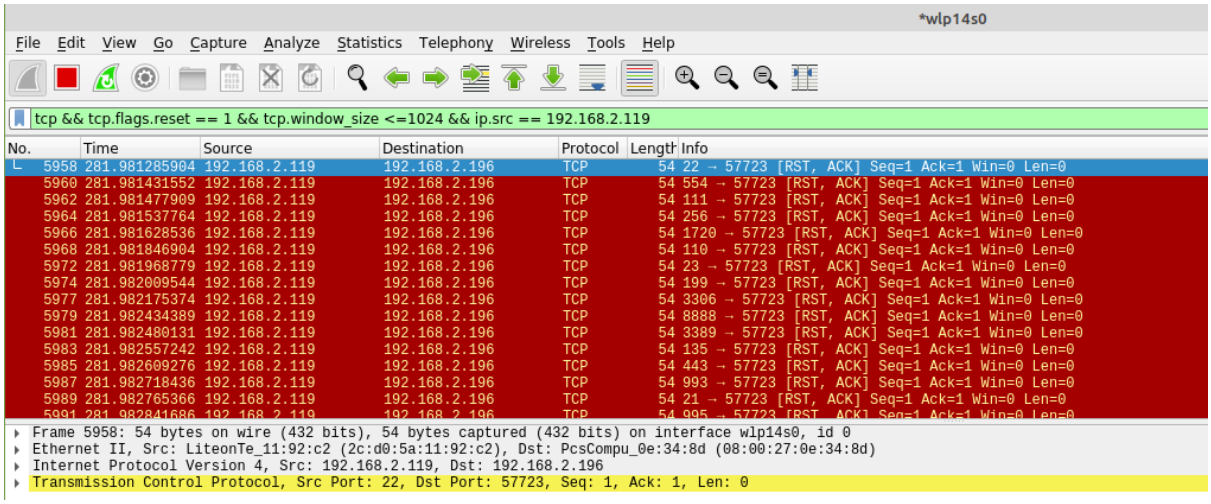


Figure 16: “Wireshark filtres d’utilitat”