# T2. Implantación y puesta en producción de sistemas IDS/IPS

# Apartados del BOE

* Análisis previo de los servicios, protocolos, zonas y equipos que utiliza la organización para sus procesos de negocio.
* Definición de políticas de corte de intentos de intrusión en los IDS/IPS
* Análisis de los eventos registrados por el IDS/IPS para determinar falsos positivos y caracterizarlos en las políticas de corte del IDS/IPS
* Relación de los registros de auditoría del IDS/IPS necesarios para monitorizar y supervisar su correcto funcionamiento y los eventos de intentos de intrusión
* Establecimiento de los niveles requeridos de actualización, monitorización y pruebas del IDS/IPS

# SNORT

* Snort es un sistema para la prevención y detección de intrusiones en la red (IPS / IDS) desarrollado por SOURCEfire y ahora por CISCO https://snort.org/prod°ucts

* Tiene su propia certificación SnortCP: http://www.sourcefire.com/products/open-source/snort-scholarship

* Para descargarlo https://snort.org/downloads

* Snort como **IDS** está basado en la red (**NIDS**)
* Implementa un motor de detección de ataques y barrido que permite:
  * **Registrar**
  * **Alertar**
  * **Responder**
* Ante cualquier anomalía previamente definida como patrones que corresponden a ataques, barridos, intentos de aprovechar alguna vulnerabilidad, **análisis de protocolos**, etc
* **TODO ESTO EN TIEMPO REAL**

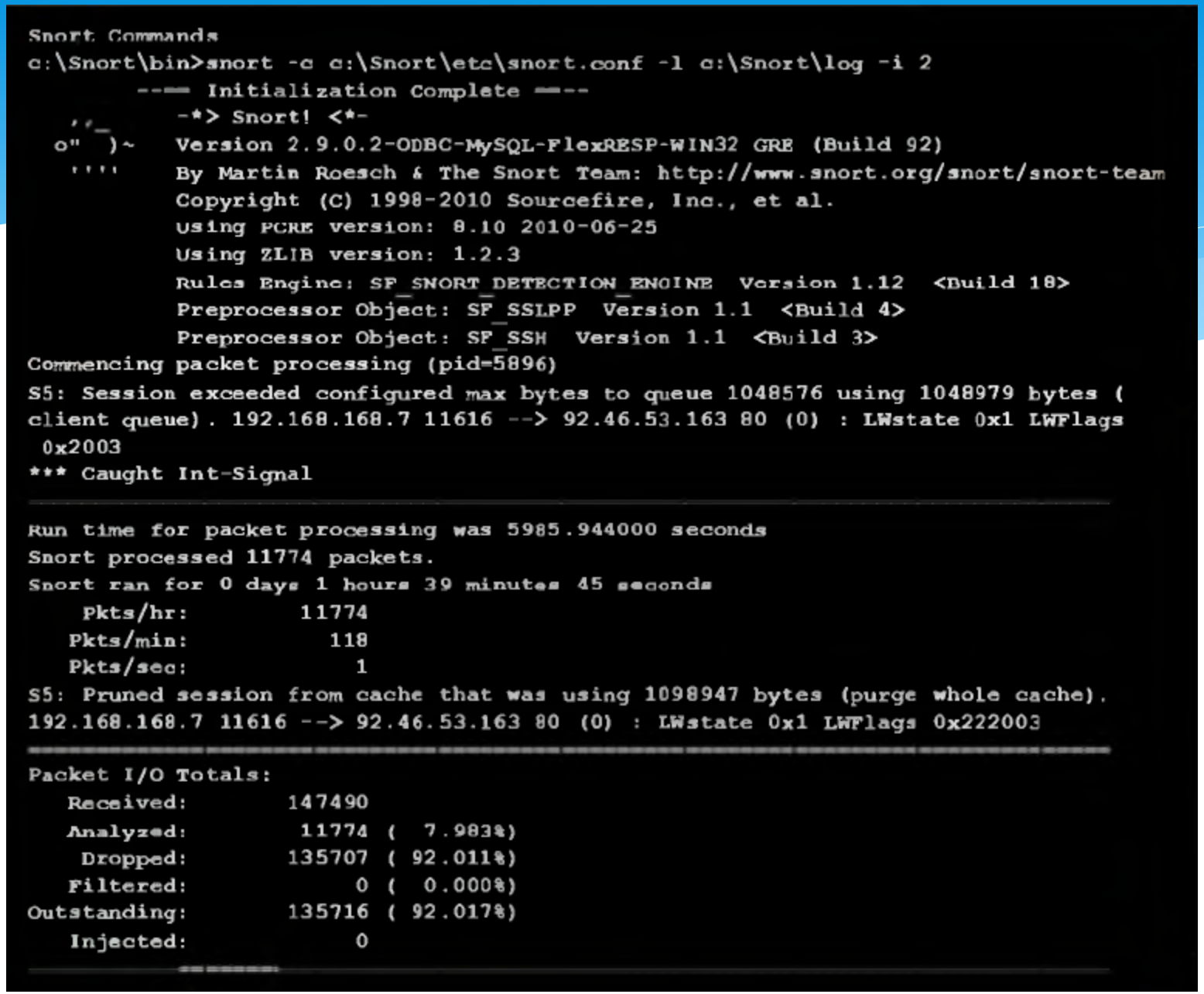
# Características de Snort

* Snort es un sistema de detección y prevención de intrusiones de red de código abierto capaz de realizar **análisis de tráfico en tiempo real** y registro de paquetes sobre redes IP.
* Puede realizar análisis de protocolos y **buscar / comparar su contenido.** Puede ser utilizado para detectar una gran variedad de ataques y sondas, tales como
  * desbordamientos de búfer
  * escaneo de puertos
  * ataques CGI
  * examinaciones SMB
  * OS fingerprinting

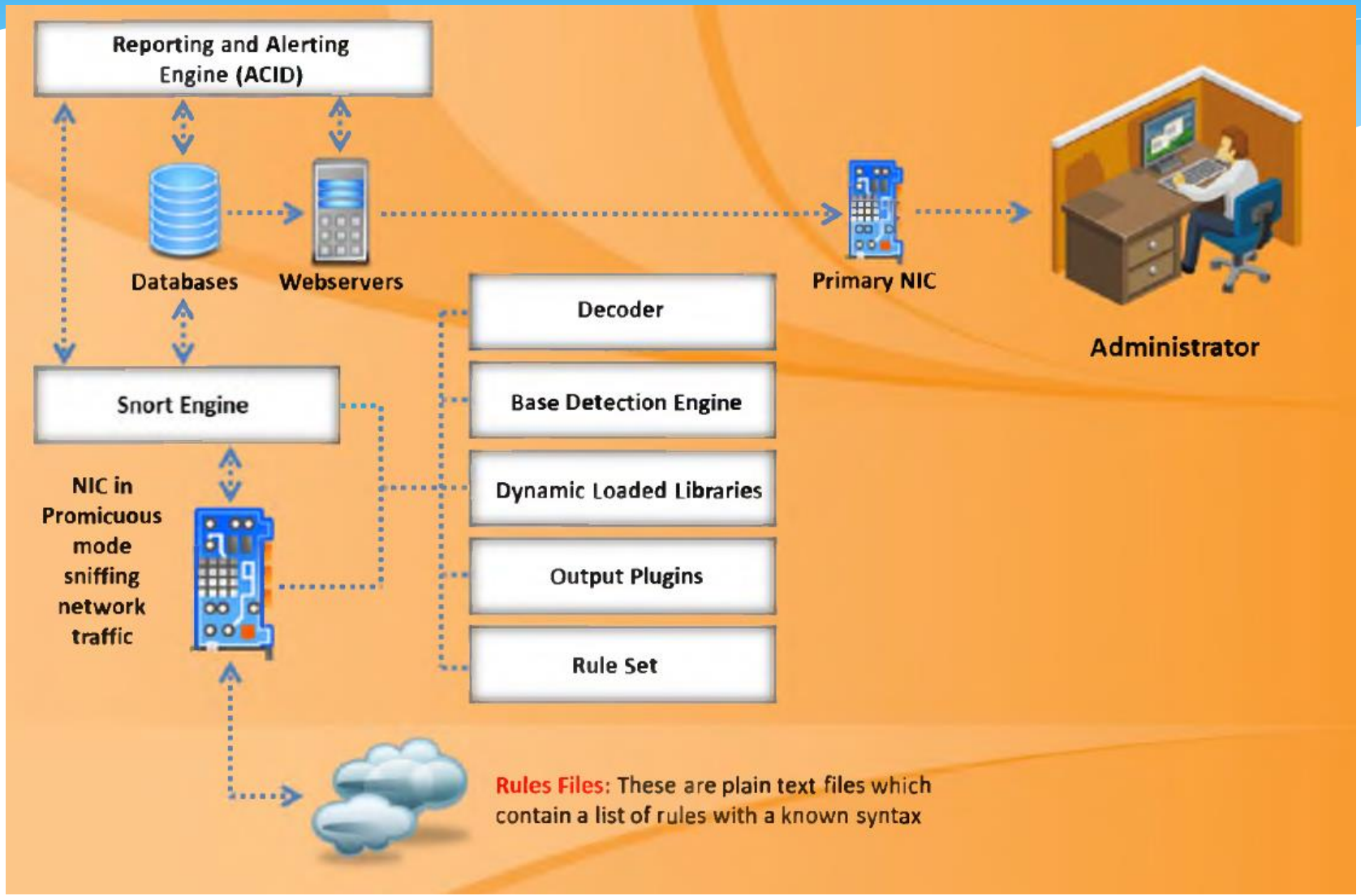
* Snort utiliza un lenguaje de **reglas flexibles** para describir el tráfico que debería recoger o dejar pasar, así como un motor de detección que utiliza una **arquitectura modular tipo plug-in**

* Snort tiene la capacidad de alertar, estos mecanismos de alertas se pueden aplicar a:
  * syslog en tiempo real
  * un archivo especificado por el usuario
  * un socket UNIX
  * mensajes WinPopup a clientes Windows.

* Snort tiene tres usos principales:
  * un **analizador de paquetes** directamente tipo tcpdump
  * un **log de paquetes** (útil para la red de depuración de tráfico, etc.)
  * **un sistema de prevención de intrusiones de red** en toda regla (full-blown)


Snort Commands
c:\Snort\bin>snort -c c:\Snort\etc\snort.conf -l c:\Snort\log -i 2
--== Initialization Complete ==--
-*> Snort! <*-
Version 2.9.0.2-ODBC-MySQL-FlexRESP-WIN32 GRE (Build 92)
By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
Copyright (C) 1998-2010 Sourcefire, Inc., et al.
Using PCRE version: 8.10 2010-06-25
Using ZLIB version: 1.2.3
Rules Engine: SF_SNORT_DETECTION_ENGINE Version 1.12 <Build 18>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Commencing packet processing (pid=5896)
S5: Session exceeded configured max bytes to queue 1048576 using 1048979 bytes (
client queue). 192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags
0x2003
*** Caught Int-Signal
Run time for packet processing was 5985.944000 seconds
Snort processed 11774 packets.
Snort ran for 0 days 1 hours 39 minutes 45 seconds
Pkts/hr: 11774
Pkts/min: 118
Pkts/sec: 1
S5: Pruned session from cache that was using 1098947 bytes (purge whole cache).
192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags 0x222003
Packet I/O Totals:
Received: 147490
Analyzed: 11774 ( 7.983%)
Dropped: 135707 ( 92.011%)
Filtered: 0 ( 0.000%)
Outstanding: 135716 ( 92.017%)
Injected: 0

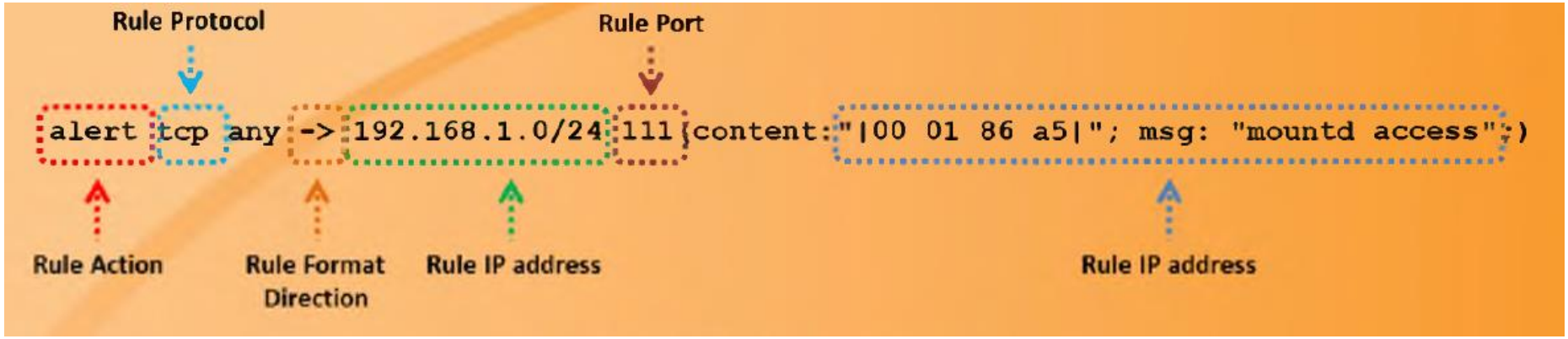
# ¿Cómo funciona el Snort?

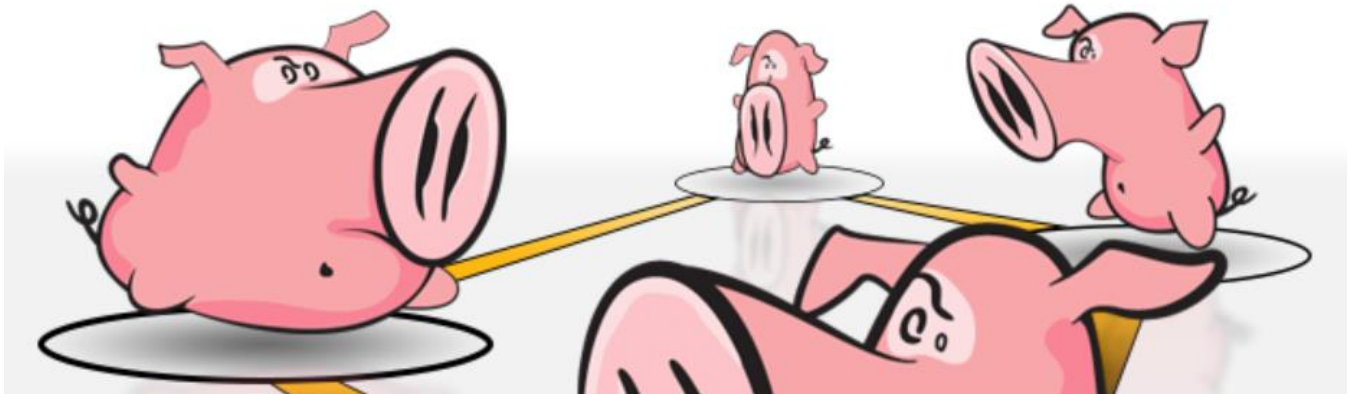


* **Decodificador**
  * Guarda los paquetes capturados, identifica los protocolos a nivel de enlace y decodifica IP
* **Motor de detección**
  * Comprueba los paquetes con las reglas previamente cargadas en la inicialización del Snort
* **Salida por diferentes Plug-ins**
  * Estos módulos permite cambiar el formato de las notificaciones al usuario, por ejemplo:
    * Consola
    * Archivos externos
    * Bases de datos

# Reglas en Snort

* Ejemplo de una regla en Snort:



# o

* Snort utiliza una popular biblioteca libpcap (para UNIX / Linux) o Winpcap (para Windows), la misma biblioteca que tcpdump utiliza para realizar su detección de paquetes.

* Snort decodifica todos los paquetes que pasan a través de la red y para ello la tarjeta de red tiene que estar en modo promiscuo.

* Basándose en el contenido de los paquetes de forma individual y en las reglas definidas en el archivo de configuración, se generan las alertas

# Ejemplo 2

alert tcp $EXTERNAL_NET any -> $HOME_NET any / (msg:"Escaneo ping con nmap";flags:A;ack:0; / reference:arachnids,28;classtype:attempted-recon; sid:628;/ rev:1;)

Analicemos esta alerta:

CABECERA

Acción de la regla: alert

Protocolo: tcp

Direccion IP origen: $EXTERNAL_NET (toda la red)

Puerto IP origen: any (cualquiera)

Direccion IP destino: $HOME_NET (toda nuestra red)

Puerto IP destino: any (cualquiera)

Dirección de la operación: -> (puede ser ->, <-, )

OPCIONES

Mensaje: msg

Opciones: flags:A;ack:0; reference:arachnids..(1)

Un poco de teoría:

flags:A Establece el contenido de los flags o banderas TCP, en este caso ACK (puede tener varios valores y operadores que veremos más adelante).

ack:0 Caso particular para valor ACK=0, es el valor que pone nmap para TCP ping scan.

reference:arachnids,28 Referencia un a un Advisory, alerta tipo Bugtrac, etc.

classtype:attempted-recon Categoría de la alerta según unos niveles predefinidos y prioridades (veremos más adelante las categorías).

sid:628 Identificación única para esta regla snort según unos tramos determinados.

rev:1 Identificación de la revisión o versión de la regla.

# Reglas en Snort (1/3)
# Acciones de las reglas y Protocolos IP

* El primer elemento de una regla es la acción de la regla.
* La **acción de la regla** dice Snort "**qué hacer**" cuando se encuentra un **paquete** que **coincide** con los **criterios de la regla**.
* Hay cinco acciones por defecto disponibles en Snort: **alert**, **log**, **pass**, **activate**, y **dynamic**
* Además, si está ejecutando Snort en modo en **línea de comandos**, tiene opciones adicionales que incluyen: **drop**, **reject**, and **sdrop**

* **alert** → Se genera una alerta y se guarda (log) el paquete
* **log** → guardar el paquete o no (logs)
* **pass** → ignorar el paquete
* **activate** → alerta y se activa otra regla dinámica
* **dynamic** → permanece inactiva hasta que sea activada por una regla de tipo activate

* **drop** → bloquea y guarda los paquetes
* **eeject** → bloquea, guarda los paquetes y envía paquetes TCP e ICMP para intentar reiniciar la comunicación
* **sdrop** → bloquea el paquete pero no se guarda

* Los 3 **protocolos IP** que suporta Snort para comprobar un comportamiento sospechoso son:
  * TCP
  * UDP
  * ICMP

# Reglas en Snort (2/3)
# Operador dirección y Direcciones IP

* **Operador dirección**
  * Este operador indica la dirección del tráfico que queremos controlar, puede ser:
    * una única dirección **->**
    * Bidireccional **<>**
    * NO EXISTE **<-**

```
log !192.168.1.0/24 any <> 192.168.1.0/24 23
```

* **¿Para que podemos utilizar el bidireccional?**

* **Direcciones IP**
  * Identifica las direcciones IP y los puertos en los que vamos a aplicar las reglas
  * Si se usa la palabra clave **any**, quiere decir cualquier dirección IP
  * Se puede utilizar en la regla CID netmask → ***Classless Inter-Domain Routing*** o **CIDR** «enrutamiento entre dominios sin clases»

| | | | |
|---|---|---|---|
| /24 | 1 C | 256 | 255.255.255.0 |

```
alert tcp !192.168.1.0/24 any -> 192.168.1.0/24 111 (content:
"|00 01 86 a5|"; msg: "external mountd access";)
```

  * Para especificar una **lista de IP** [192.168.1.1, 192.168.1.2, 192.168.1.16]
  * Permite el operador **NOT (!)**

# Snort Rules: Port Numbers

Port numbers can be listed in different ways, including "any" ports, static port definitions, port ranges, and by negation

Port ranges are indicated with the range operator ":"

Example of a Port Negation

```
log tcp any any -> 192.168.1.0/24 !6000:6010
```

| Protocols | IP address | Action |
|---|---|---|
| Log UDP any any -> | 92.168.1.0/24 1:1024 | Log UDP traffic coming from any port and destination ports ranging from 1 to 1024 |
| Log TCP any any -> | 192.168.1.0/24 :5000 | Log TCP traffic from any port going to ports less than or equal to 5000 |
| Log TCP any :1024 -> | 192.168.1.0/24 400: | Log TCP traffic from the well known ports and going to ports greater than or equal to 400 |

# Reglas en Snort (3/3)
# Números de puerto

* Los números de puertos pueden ser especificados de varias formas:
  * Cualquier puerto **any**
  * Puertos estáticos **80** http, **23** telnet
  * Rangos **(:)**
  * Negación **(!)** `log tcp any any -> 192.168.1.0/24 !6000:6010`

| Protocols | IP address | Action |
|---|---|---|
| Log UDP any any -> | 92.168.1.0/24 1:1024 | Log UDP traffic coming from any port and destination ports ranging from 1 to 1024 |
| Log TCP any any -> | 192.168.1.0/24 :5000 | Log TCP traffic from any port going to ports less than or equal to 5000 |
| Log TCP any :1024 -> | 192.168.1.0/24 400: | Log TCP traffic from privileged ports less than or equal to 1024 going to ports greater than or equal to 400 |