Snort & Nmap

Mike O’Connor

Eric Tallman

Matt Yasiejko

Overview

- Snort
  - What is it?
  - What does it do?
  - Features
- Nmap
  - What is it?
  - What does it do?
  - Features

What is Snort?

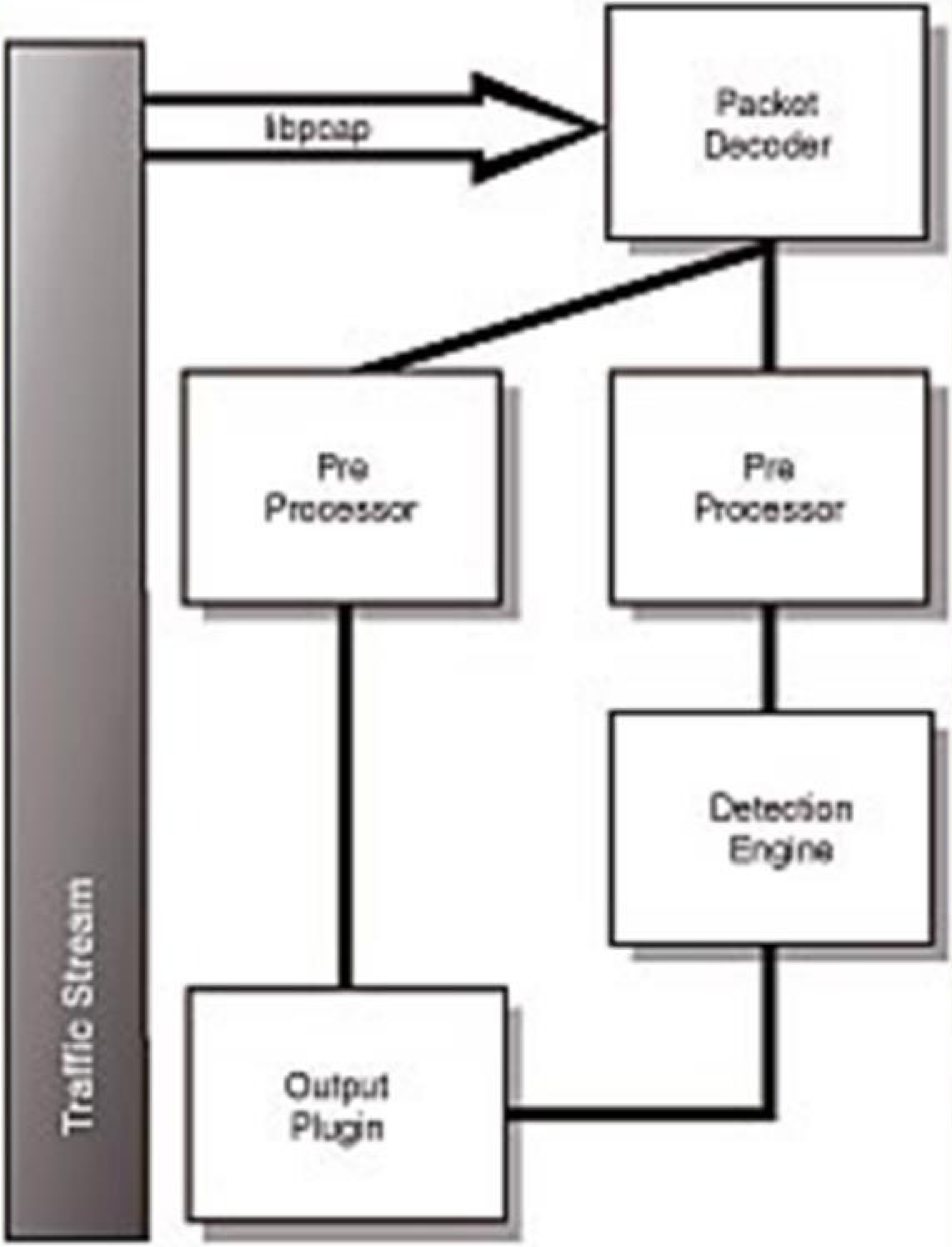
- IDS
- Can also be configured to be an IPS
- Software solution to IDS/IPS
- To be IPS, the sniffing machine needs 2 interfaces
- Network based
  - Switch – port mirroring
  - Hub – sniff all

Snort

- Network intrusion detection system
- Real-time traffic analysis
- Packet logging
- Detects OS fingerprinting attempts
  - Protocol implementation details

Components in Snort

- External packet – capture library
- Packet decoder – translates protocol elements into an internal data structure
- Preprocessors – examine/manipulate packets for detection engine
- Detection engine – tests single elements of packets
- Output plugins – generates alerts



Traffic Stream

libpcap

Packet Decoder

Pre Processor

Pre Processor

Detection Engine

Output Plugin

1. Capturing traffic (libpcap/WinPcap)

- Sniffs line and gets <u>raw</u> packets off the network
- Raw packets needed to detect various attacks
- Can only process one packet at a time

- We use WinPcap → Windows Packet Capturing
  - Captures packets traveling across a network

2. Packet decoder

- Series of decoders that each decode specific protocol elements
- Data structure is filled up with decoded packet data
- Data structures passed to preprocessors and the detection engine

3a. Preprocessors

- Two types
  - Examine packets

    -Used for non-signature based attacks
  - Modify packets in preparation for detection engine

    -Normalize traffic
- Packets cycle through all preprocessors
  - Keeps attackers from hiding other traffic
  - Multiple violations may be seen this way

3b. Preprocessors

- Fragmentation
  - Malicious traffic
    - Modify packet headers
    - DoS – Ping of Death
- Stateful inspections
- Stateless connections
  - SYN-ACK (connection not complete)
- IP protocol checks – beyond TCP

4. Detection engine

- Uses a decision tree
  - Eg) if the packet is TCP, the packet is passed to the portion that deals with TCP
  - The first signature that matches is applied, the next packet is analyzed
    - Priority is very important
    - High level attacks must be prioritized currently

5. Output plugins

- Dumps alert data to a file/resource
- Unified format
  - One of many options
  - Fastest possible
    - Alert file – Attack summary, IPs, protocol used, etc listed
    - Packet file – actual packet info
- Database, file dumps, external applications

snort_inline turns Snort into IPS

- Set up rules to drop packets
- Set up alerts to log attacks
- Set up rules to cut connection
  - TCP reset for example
- drop tcp any any -> any 80 (classtype:attempted-user; msg:"Port 80 connection initiated";)

General rule structure

- _action _protocol _ip1 _direction _ip2 (options)

_action options

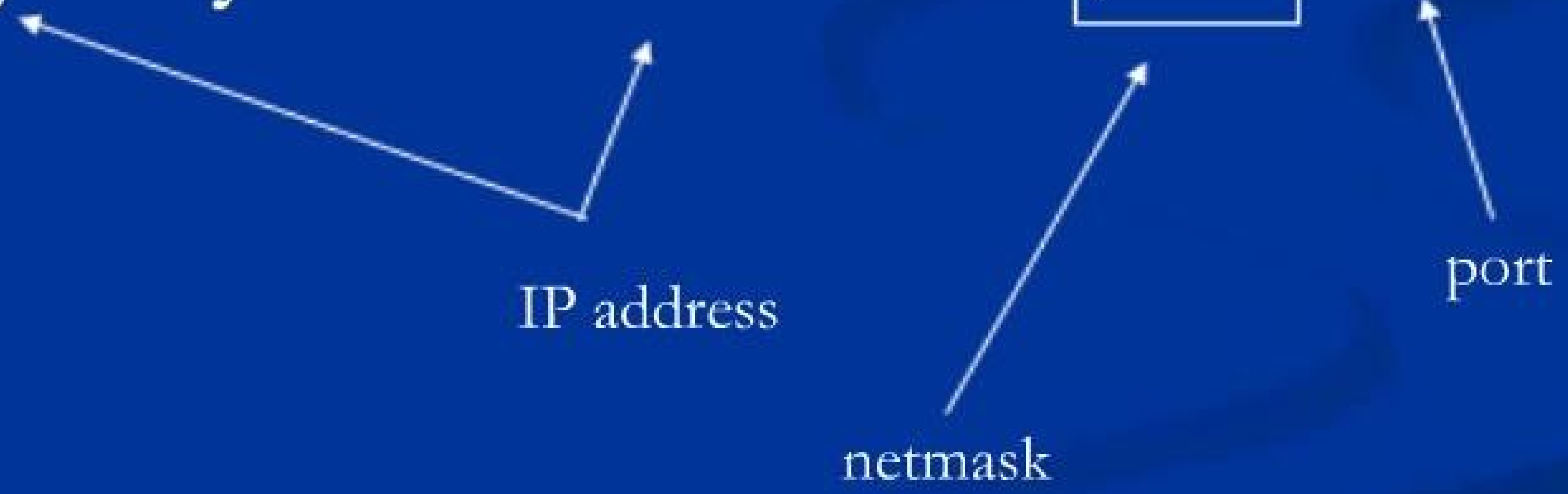
- **_action** _protocol _ip1 _direction _ip2 (options)
- alert - generate an alert using the selected alert method, and then log the packet
- log - log the packet
- pass - ignore the packet
- activate - alert and then turn on another dynamic rule
- dynamic - remain idle until activated by an activate rule , then act as a log rule

_protocol options

- _action _protocol _ip1 _direction _ip2 (options)
- TCP, IP, UDP, ICMP (, ARP, IGRP, GRE, OSPF, RIP, IPX)

_ip options

- _action _protocol _ip1 _direction _ip2 (options)
- IP address/netmask, port, ! to negate
  - Any, individual ip
- alert tcp any any -> 192.168.1.0/24 111



IP address

port

netmask

_direction options

- _action _protocol _ip1 _direction _ip2 (options)

- -> is from source to destination
- <> is from source to destination and destination to source

Rule options

- _action _protocol _ip1 _direction _ip2 (options)
- alert tcp any any -> $HOME_NET 31337 (msg: "BLEEDING-EDGE ATTACK RESPONSE Potential root shell connection detected!"; flow: established,to_server; tag: session, 20, packets; classtype: bad-unknown; sid: 2001545; rev:2;)

Rule structure for wireless

<action> wifi <mac> <direction> <mac> (<rule options>)

<MAC address> Rule options

- # Single MAC Address 00:DE:AD:BE:EF:00
- # MAC Address List [00:DE:AD:BE:EF:00, 00:DE:AD:C0:DE:00, ...]

Logs

- Using syslog logs
- Sawmill
  - Logs need to be converted to plaintext to be processed
  - Web interface to analyze traffic
  - Windump -r _log_ -tt > _txtFile_

Snort Status

- DB connection is problematic for FreeBSD version
- Snort currently captures traffic and creates logs based on rules
- Lab3 is now the sniffer box
  - WinPcap and Snort
- Plugged into physical port FA0/23
  - Receiving all switch traffic

NMAP

Nmap

- Network Mapper
- Discovers services available on different hosts in a network
- Command line, GUI versions
  - Nmap and nmapfe packages in FreeBSD

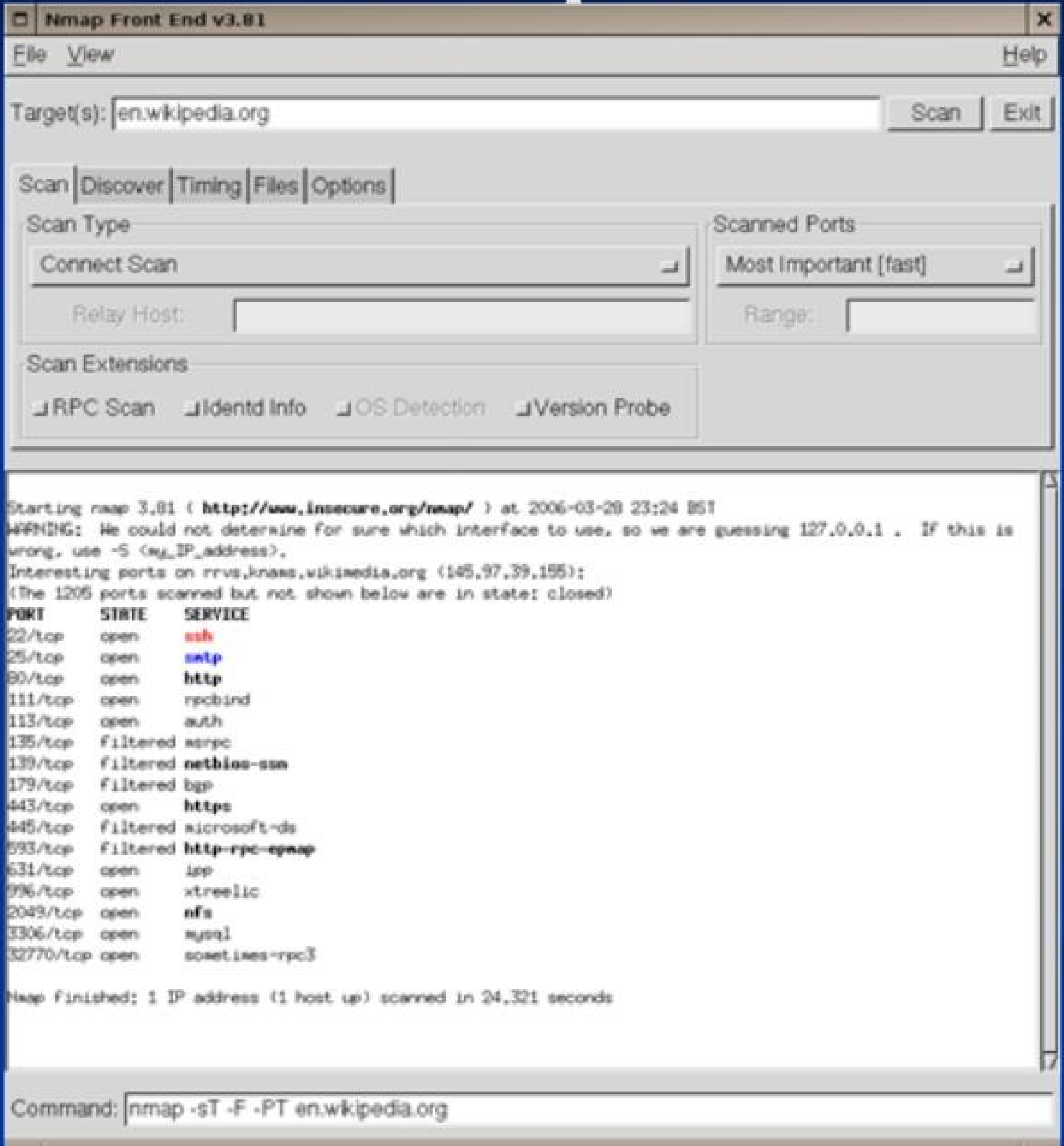
Features

- Enumerates ports on target machines
- Identify services running on those ports
- OS fingerprinting

Typical uses

- List services available on a machine
- Run network security audit of machines
- Identify computers that may be exploited
- Audit individual machine security

nmapfe



Target(s): en.wikipedia.org

Scan Type: Connect Scan

Scanned Ports: Most Important [fast]

Scan Extensions: □ RPC Scan □ Identd Info □ OS Detection □ Version Probe

```
Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2006-03-28 23:24 BST
WARNING:  We could not determine for sure which interface to use, so we are guessing 127.0.0.1 .  If this is
wrong, use -S <my_IP_address>.
Interesting ports on rrvs.knams.wikimedia.org (145.97.39.155):
(The 1205 ports scanned but not shown below are in state: closed)
PORT      STATE    SERVICE
22/tcp    open     ssh
25/tcp    open     smtp
80/tcp    open     http
111/tcp   open     rpcbind
113/tcp   open     auth
135/tcp   filtered msrpc
139/tcp   filtered netbios-ssn
179/tcp   filtered bgp
443/tcp   open     https
445/tcp   filtered microsoft-ds
593/tcp   filtered http-rpc-epmap
631/tcp   open     ipp
996/tcp   open     xtreelic
2049/tcp  open     nfs
3306/tcp  open     mysql
32770/tcp open     sometimes-rpc3

Nmap finished: 1 IP address (1 host up) scanned in 24.321 seconds
```

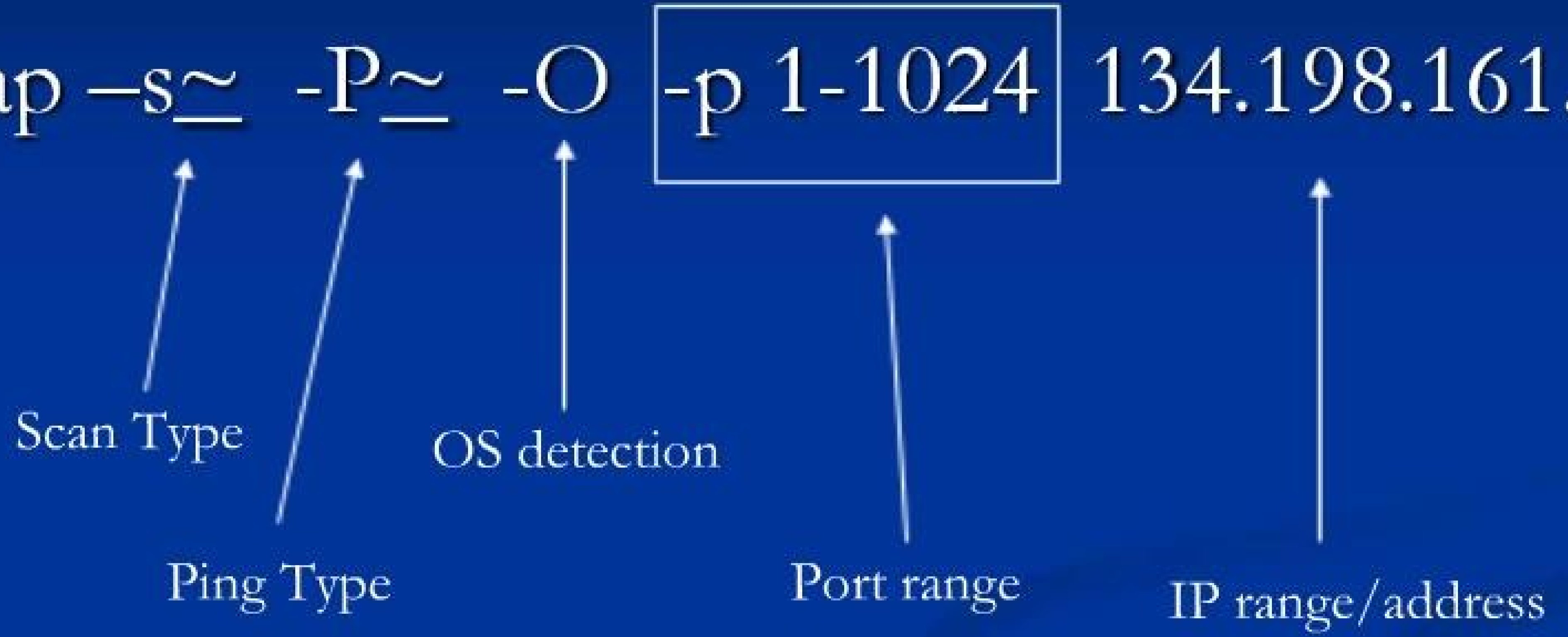
Command: nmap -sT -F -PT en.wikipedia.org

Just the beginning…

- Nmap is one tool in an arsenal for black hat hackers
- Prelude to exploitation tools
  - Metasploit - used for actual exploitation attempt

Nmap command

- nmap –s~ -P~ -O -p 1-1024 134.198.161.*



Scan Type

Ping Type

OS detection

Port range

IP range/address

Enumerate ports / services

- "Well-known" or "Interesting" ports

  - 1-1024

  - 65,535 total TCP & UDP ports

- Port/Protocol State Service Name

Types of scans

- http://www.secguru.com/nmap_cheatsheet
- sS (TCP SYN scan) – half open scan; stealthy
  - SYN/ACK – listening; RST – non-listener
- sT (TCP connect scan) – uses system call to make connection; easily logged
- sU (UDP scans) – sends empty UDP header to targeted ports; code returned indicates port state
- sN; -sF; -sX (TCP Null, FIN, and Xmas scans)
  - If SYN, RST, ACK bits not set (TCP RFC)
    - Any incoming segment not containing RST causes a closed port to respond with an RST
    - No response if port is open

OS detection

- Uses TCP/IP fingerprinting
  - OS particular implementation of protocol indicates target host OS
  - Checked against DB of known DB signatures
- Why hide OS?
  - Black hat hackers might try OS specific exploits if known

- http://www.csee.umbc.edu/~krishna/cs491n/snort_manual.pdf