1- Snort 3 primary operations

Snort has three primary operating modes:

Sniffer mode

- Sniffer mode allows Snort to capture packets off the wire and outputs them to the screen in a human readable format
- It also provide some useful traffic statistics when the capture process is stopped
- Packet sniffer mode is how Snort runs by default
- Snort can be executed in this mode by simply specifying a capture interface with the command **snort –i <interface>**

Packet logger mode

- Packet logger mode is same as sniffer mode, only it logs packets to a file rather than screen
- This data is most commonly logged in binary PCAP format
- This operation mode is enabled by command:
  - snort –l <log directory>
- Stored PCAP files can be read by command:
  - snort –r <pcap file>

NIDS mode

- NIDS mode is designed to read data captured from the network
- To do this, packet data traverses different phases of Snort's architecture shown in figure

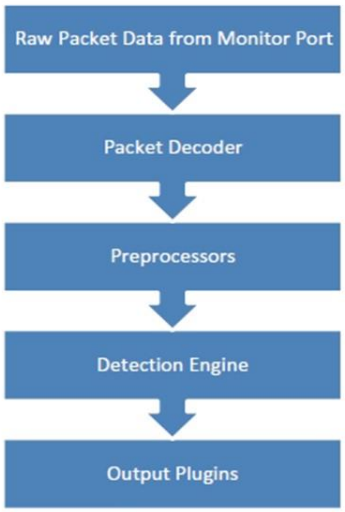

FIGURE 9.4
The Snort NIDS Mode Architecture

- Snort can receive data by parsing a manually specified PCAP file or by pulling it directly from a sensors monitoring interface

- Then it analyzes data with packet decoder, which is actually a series of multiple decoders that analyze packet data and normalize it
- After that the data is sent to Snort's preprocessors, which are of two types:
  - The first type is used for detection purposes
  - The second type modify packet data so that it can be better parsed by detection engine
- After preprocessing, data is shipped to workhorse of Snort architecture, the detection engine
- The detection engine is responsible for parsing rules and determining if those rules match the traffic being analyzed
- When the detection engine determines that network traffic matches a rule, it hands that data over to the output plugins
- As a result of which an alert is generated to notify the analyst

---

2

3

- Unified2
  - In an enterprise environment, the most commonly used log format is Unified2.
  - This is a binary format capable of storing both the alert data and the packet data associated with it.
  - Unified2 output can be read by tools like Barnyard2 or Pigsty.
  - These tools are used for interpreting Unified2 output and placing that alert data into a database.
  - Snort also includes a tool called **u2spewfoo** that is able to read the unified2 format.

4

rule header

- The rule header is always the first and required component of the rule
- It is responsible for defining "who" is involved in the traffic pattern
- Everything defined in the rule header can be found within the header of a packet
- The rule header always consists of the same parts: rule action, protocol, source/dest hosts, source/dest ports and the direction of the traffic

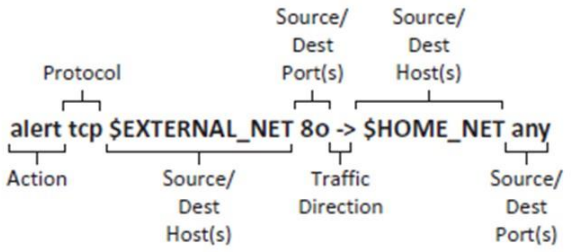

FIGURE 9.20

The IDS Rule Header

5

- Honeypots are categorized by the level of interaction they provide and are most commonly designated as either:
  - Low interaction honeypots
  - High interaction honeypots
- Organizational goals, the assets you are protecting and the services you wish to emulate will help define the level of interaction needed for your honeypot deployment

low interaction honeypots

- A low interaction honeypot is software-based, and is designed to emulate one or more services
- The level of interaction they provide is dependent upon the service being emulated and the software itself
- For instance, Kippo is a low interaction honeypot that mimics the SSH service
- It allows an attacker to log in to the service and even to browse a fake file system

High Interaction Honeypot

- A high interaction honeypot is actually configured to mirror a production system, and is designed to give an attacker full control of an operating system
- This system will be configured to utilize extensive system and file system logging
- These honeypots often exist as virtual machines so that they can be reverted back to a known clean snapshot with relative ease

---

6

Honeyd

- When someone brings up the history of honeypot software, the discussion typically begins with Honeyd
- The Honeyd utility was developed by Niels Provos over ten years ago and provides ability to emulate hosts as a low-interaction honeypot for Windows and LINUX environemnt
- Honeyd has been the de facto low-interaction honeypot solution for years and is so popular that a lot of modern honeypot solutions borrow from its functionality
- The best way to show the functionality of Honeyd is to demonstrate it in practice

- Installing Honeyd can be accomplished via following command:
  - apt-get install honeyd
- To run Honeyd, we have to create a configuration file
- A default configuration file that contains a few different examples of how the file is structured is provided at /etc/honeypot/honeyd.conf
- Honeyd scripts that can emulate a variety of services SMTP, SNMP, TELNET, POP, and other services.

Kippo SSH Honeypot

- Kippo is a low interaction honeypot that simulates an SSH server
- It is designed to detect brute force attempts and log attacker interaction with a simulated shell environment
- Kippo is useful as a canary honeypot because SSH protocol is commonly used to manage both Unix-based and network devices
- When an attacker gains a foothold onto the network, a couple of scenarios exist where the attacker might try to access devices using the SSH service:
  - The attacker will attempt a brute force or dictionary attack against the SSH server to gain access
  - The attacker will attempt to guess password of a user to gain access
  - The attacker will attempt to log into the service with credentials already obtained through some other means

---

7

BPF

- The BPF syntax is the most commonly used packet filtering syntax and is used by a number of packet processing applications
- Tcpdump uses BPF syntax exclusively and Wireshark and tshark can use BPF syntax while capturing packets from the network
- BPFs can be used during collection in order to eliminate unwanted traffic, or they can be used while analyzing traffic that has already been collected by a sensor
- A filter created using the BPF syntax is called an expression
- These expressions have a particular anatomy and structure, consisting of one or more primitives that can be combined with operators
- A primitive can be thought of as a single filtering statement and they consist of one or more qualifiers, followed by a value in the form of an ID name or number

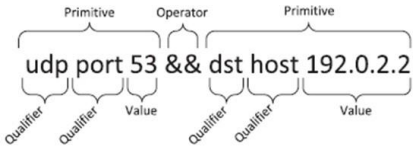

FIGURE 13.42

A Sample BPF Expression

Table 13.1 BPF Qualifiers

Qualifier Type	Qualifier	Description
Type		Identifies what the value refers to. "What are you looking for?"
	`host`	Specify a host by IP address
	`net`	Specify a network in CIDR notation
	`port`	Specify a port
Dir		Identifies the transfer direction to or from the value. "What direction is it going?"
	`src`	Identify a value as the communication source
	`dst`	Identify a value as the communication destination
Proto		Identifies the protocol in use. "What protocol is it using?"
	`ip`	Specify the IP protocol
	`tcp`	Specify the TCP protocol
	`udp`	Specify the UDP protocol

Table 13.2 BPF Logical Operators

Operator	Symbol	Description
Concatenation (AND)	`&&`	Evaluates to true when both conditions are true
Alternation Operator (OR)	`||`	Evaluates to true when either condition is true
Negation Operator (NOT)	`!`	Evaluates to true when a condition is NOT met

Table 13.3 Example BPF Expressions

Expression	Description
`host 192.0.2.100`	Matches traffic to or from the IPv4 address specified
`dst host 2001:db8:85a3::8a2e:370:7334`	Matches traffic to the IPv6 address specified
`ether host 00:1a:a0:52:e2:a0`	Matches traffic to the MAC address specified
`port 53`	Matches traffic to or from port 53 (DNS)
`tcp port 53`	Matches traffic to or from TCP port 53 (Large DNS responses and zone transfers)
`!port 22`	Matches any traffic not to or from port 22 (SSH)
`icmp`	Matches all ICMP traffic
`!ip6`	Matches everything that is not IPv6

Generating threat intelligence

- This type of threat intelligence can be broken down into three sub categories:
  - **Strategic**
- Strategic Intelligence is information related to the strategy, policy and plans of an attacker at a high level.
- By government or military organizations in response to threats from other governments or militaries.
- Larger organizations are now developing these capabilities and sell strategic intelligence as a service.
- This is focused on the long-term goals of the force supporting the individual attacker or unit.
- Artifacts of this type of intelligence can include policy documents, war doctrine, position statements, and government, military, or group objectives.
  - **Operational**
- Operational Intelligence is information related to how an attacker or group of attackers plans and supports the operations that support strategic objectives.
- This is different from strategic intelligence because it focuses on narrower goals, often more timed for short-term objectives that are only a part of the big picture.
- Some public organizations will have visibility into these attacks, with an ability to generate operational intelligence.

- Artifacts of this type of intelligence are similar, but often more focused versions of artifacts used for the creation of strategic intelligence.
    - **Tactical**
- Tactical Intelligence refers to the information regarding specific actions taken in conducting operations at the mission or task level.
- Here we dive into the tools, tactics, and procedures used by an attacker, and where 99% of SOCs performing NSM will focus their efforts.
- The individual actions of an attacker or group of attackers are analyzed and collected.
- This often includes artifacts such as indicators of compromise (IP addresses, file names, text strings) or listings of attacker specific tools.
- This intelligence is the most transient and becomes outdated quickly.
- The threat will typically begin as an IP address that shows up in an IDS alert
- Or it may manifest as a suspicious file downloaded by a client.
- Tactical threat intelligence is generated by researching this data and tying it together in an investigation.