

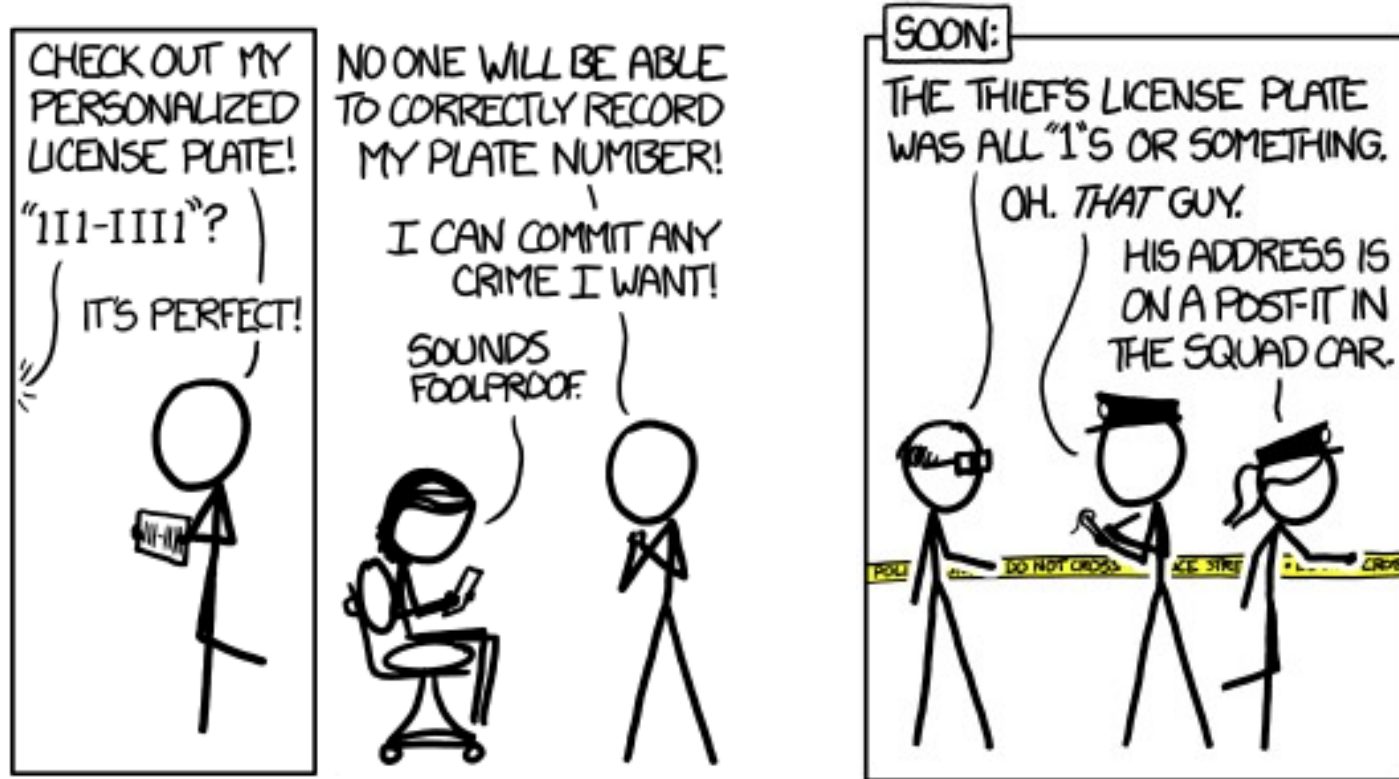
# IDS and IPS Systems



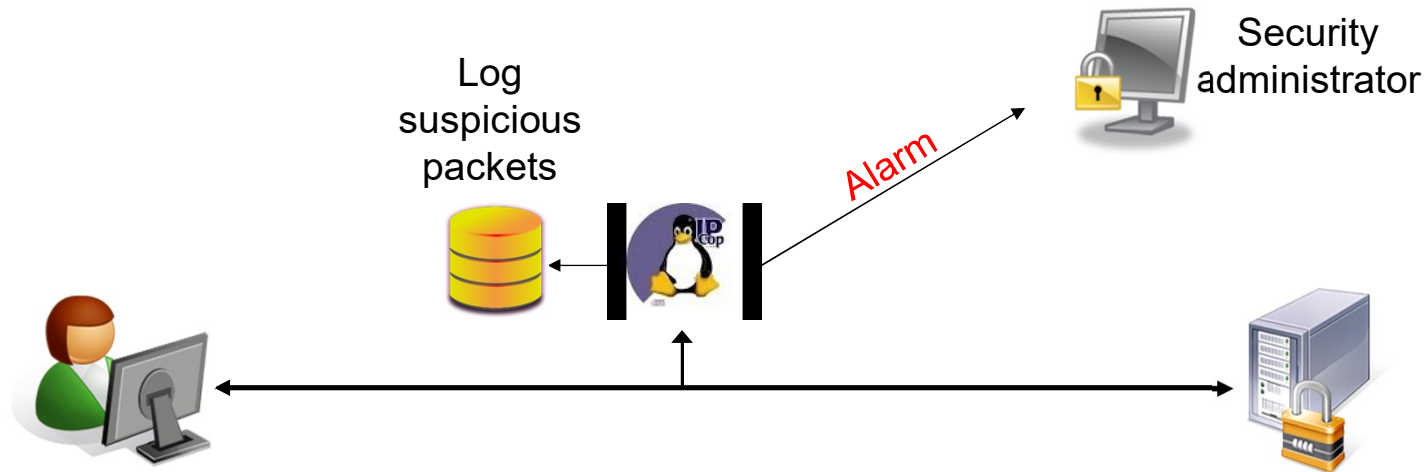
Intrusion detection and prevention systems

Chapter 21.2

# How to hide activities – evading IDS systems



# Intrusion Detection System (IDS)



An **IDS** is passive and sends alarms when rules are triggered

An **IPS** (intrusion prevention system) can **take actions** and for example block packets and modify some firewall settings: “block all from 10.1.1.44” for 10 minutes. **IPS systems may enable DoS attacks!**

# Intrusion Detection System (IDS)

HIDS: **Host-based** IDS located in individual hosts (clients, servers)

NIDS: **Network-based** IDS looks at traffic on a network segment

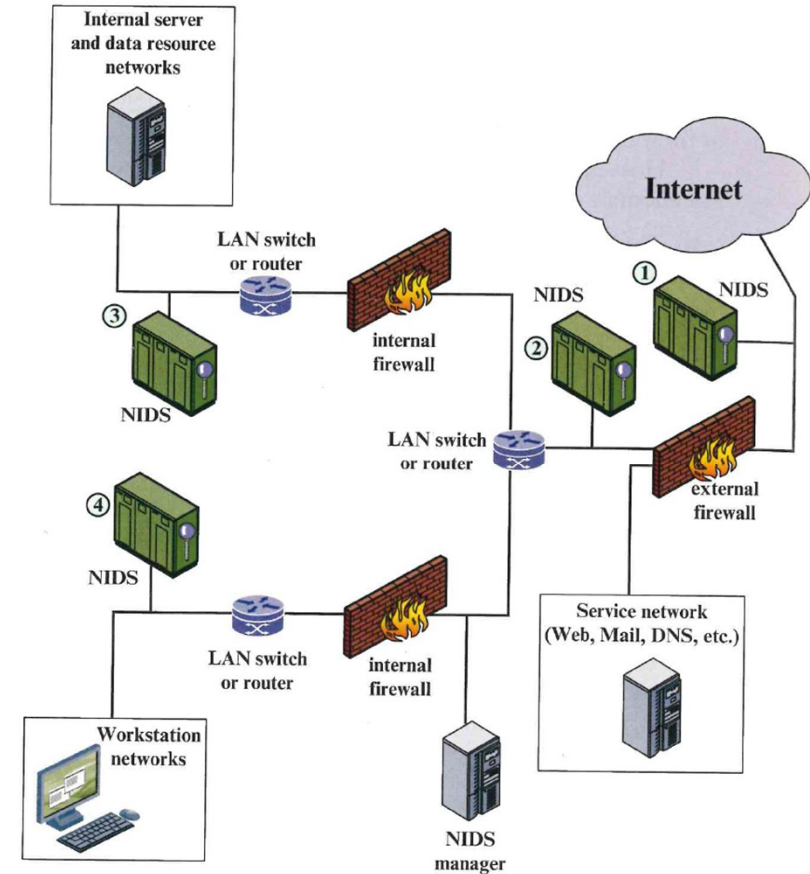
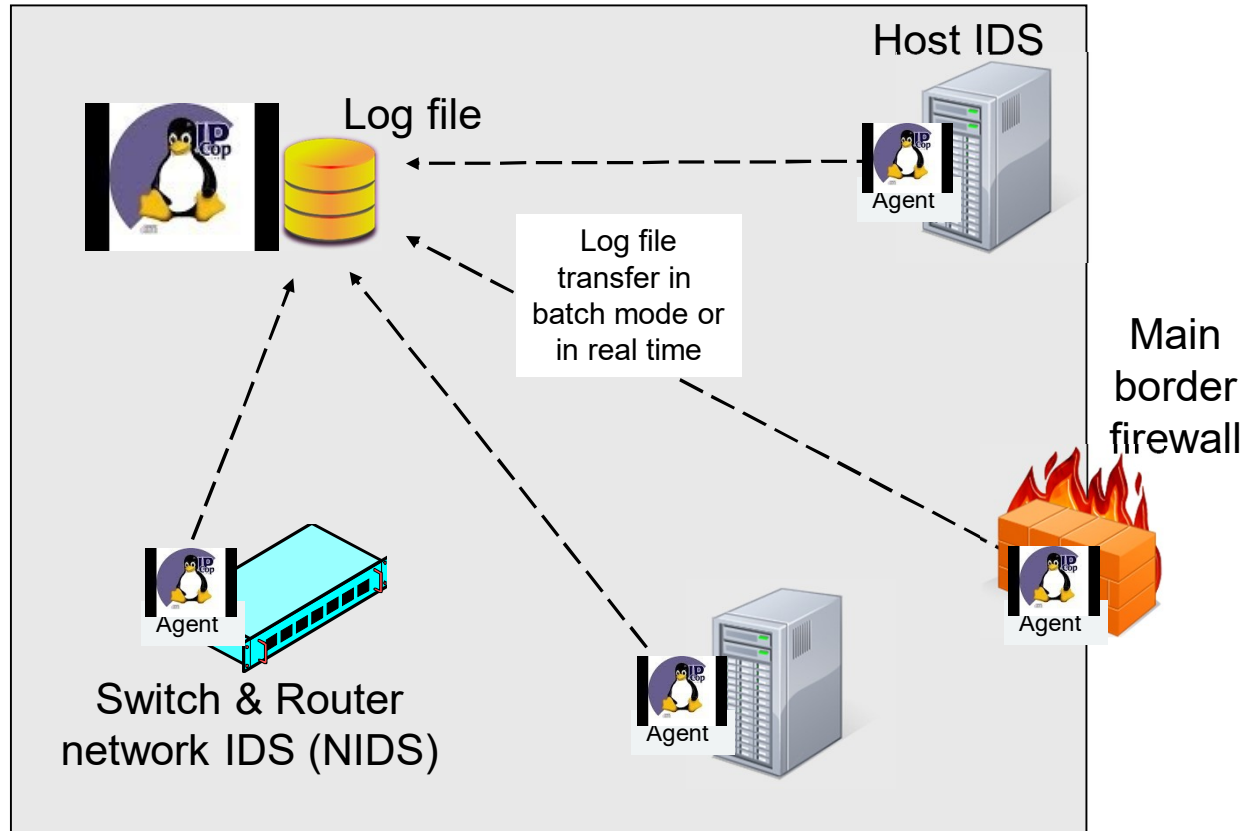
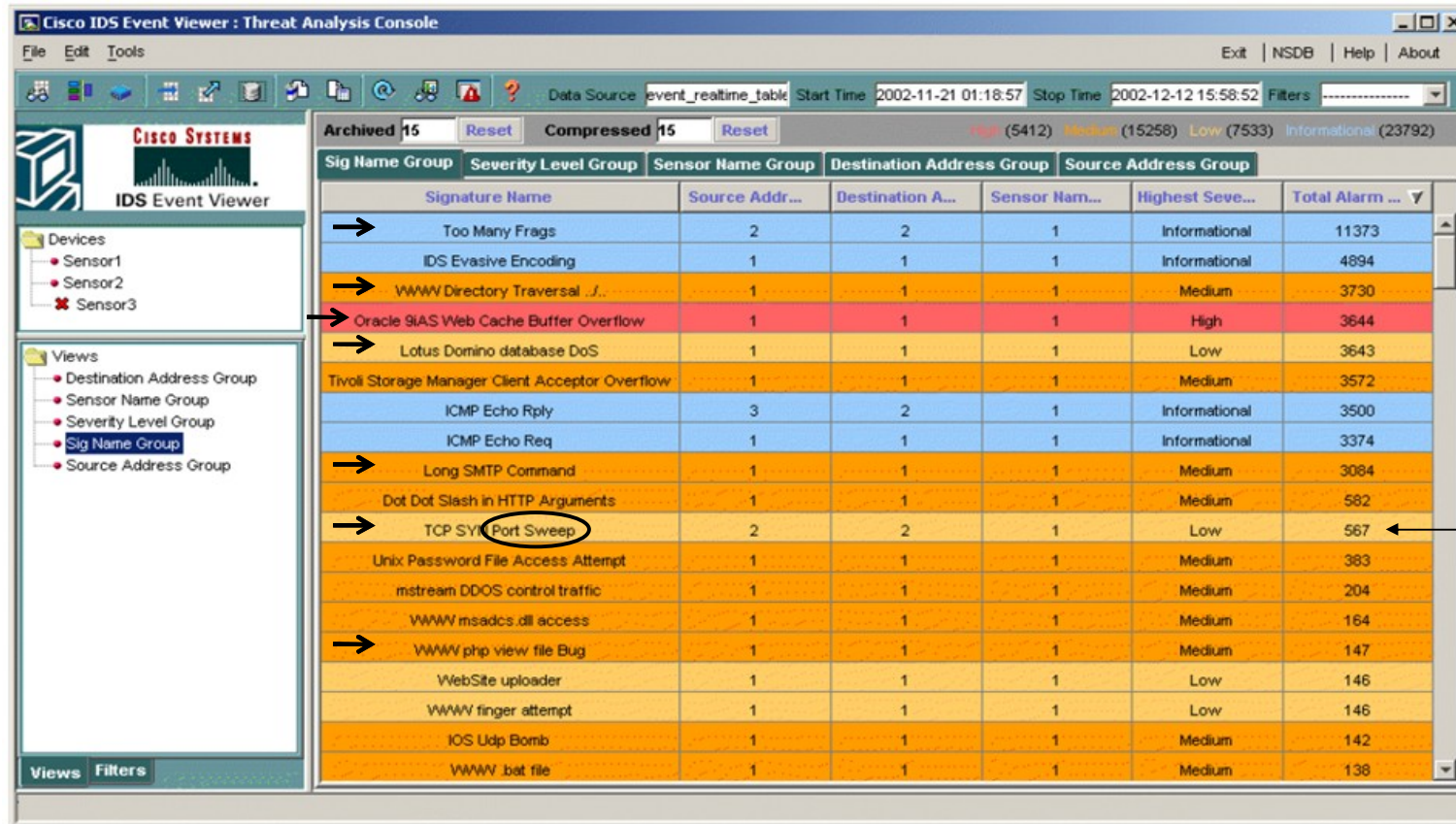


Figure 21.6 Example of NIDS Sensor Deployment

# IDS with centralized monitoring



# Intelligent logging with analysis tools



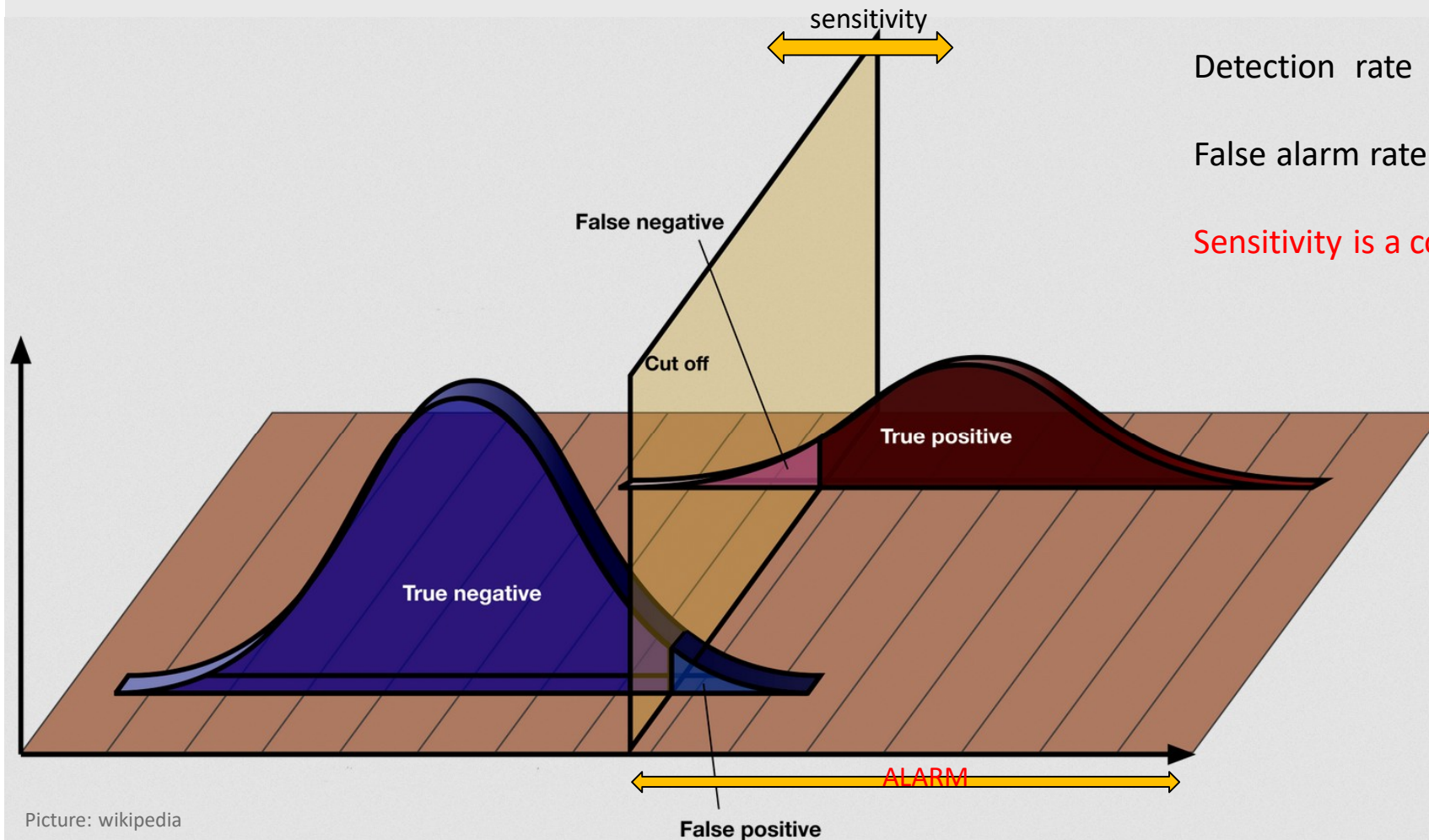
The screenshot shows the Cisco IDS Event Viewer interface. On the left, there are panels for 'Devices' (Sensor1, Sensor2, Sensor3) and 'Views' (Destination Address Group, Sensor Name Group, Severity Level Group, Sig Name Group, Source Address Group). The main area displays a table of events with columns: Signature Name, Source Addr..., Destination A..., Sensor Nam..., Highest Seve..., and Total Alarm ... . The table is filtered by 'Data Source: event\_realtime\_table' and 'Start Time: 2002-11-21 01:18:57' to 'Stop Time: 2002-12-12 15:58:52'. The table shows 15 archived and 15 compressed events. The 'TCP SYN Port Sweep' event is highlighted with a red arrow and a callout box.

Signature Name	Source Addr...	Destination A...	Sensor Nam...	Highest Seve...	Total Alarm ...
Too Many Frags	2	2	1	Informational	11373
IDS Evasive Encoding	1	1	1	Informational	4894
WWW Directory Traversal ..	1	1	1	Medium	3730
Oracle 9iAS Web Cache Buffer Overflow	1	1	1	High	3644
Lotus Domino database DoS	1	1	1	Low	3643
Tivoli Storage Manager Client Acceptor Overflow	1	1	1	Medium	3572
ICMP Echo Rply	3	2	1	Informational	3500
ICMP Echo Req	1	1	1	Informational	3374
Long SMTP Command	1	1	1	Medium	3084
Dot Dot Slash in HTTP Arguments	1	1	1	Medium	582
TCP SYN Port Sweep	2	2	1	Low	567
Unix Password File Access Attempt	1	1	1	Medium	383
mstream DDOS control traffic	1	1	1	Medium	204
WWW msadcs.dll access	1	1	1	Medium	164
WWW.php view file Bug	1	1	1	Medium	147
WebSite uploader	1	1	1	Low	146
WWW finger attempt	1	1	1	Low	146
IOS Udp Bomb	1	1	1	Medium	142
WWW .bat file	1	1	1	Medium	138

One event generated from many packets



# Sensitivity is a compromise




$$\text{Detection rate} = \frac{TP}{TP+FN}$$

$$\text{False alarm rate} = \frac{FP}{FP+TN}$$

Sensitivity is a compromise

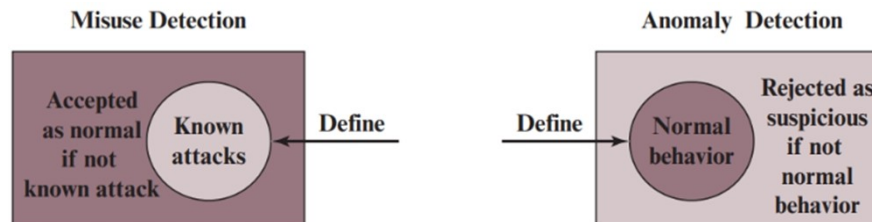
# Signature and Anomaly Based Detection

## Signature based (misuse)

- Signatures/specifications for known attacks
- Few false alarms
- Cannot stop new attacks where no signatures exist
- Cannot detect zero-day attacks targeting vulnerabilities that are not yet known
- Compare with anti-virus software
-  Free multi-platform tool owned by Cisco

## Anomaly based

- Detects anomalous, unusual, behavior
  - Traffic or behavior new to the network
  - Can look at statistical patterns
- The only way to stop zero-day attacks
- Needs training – but what is normal?
- Risk for too many false alarms
- Hard to know what it has really learned
- Compare with anti-spam software





# Snort packet processing

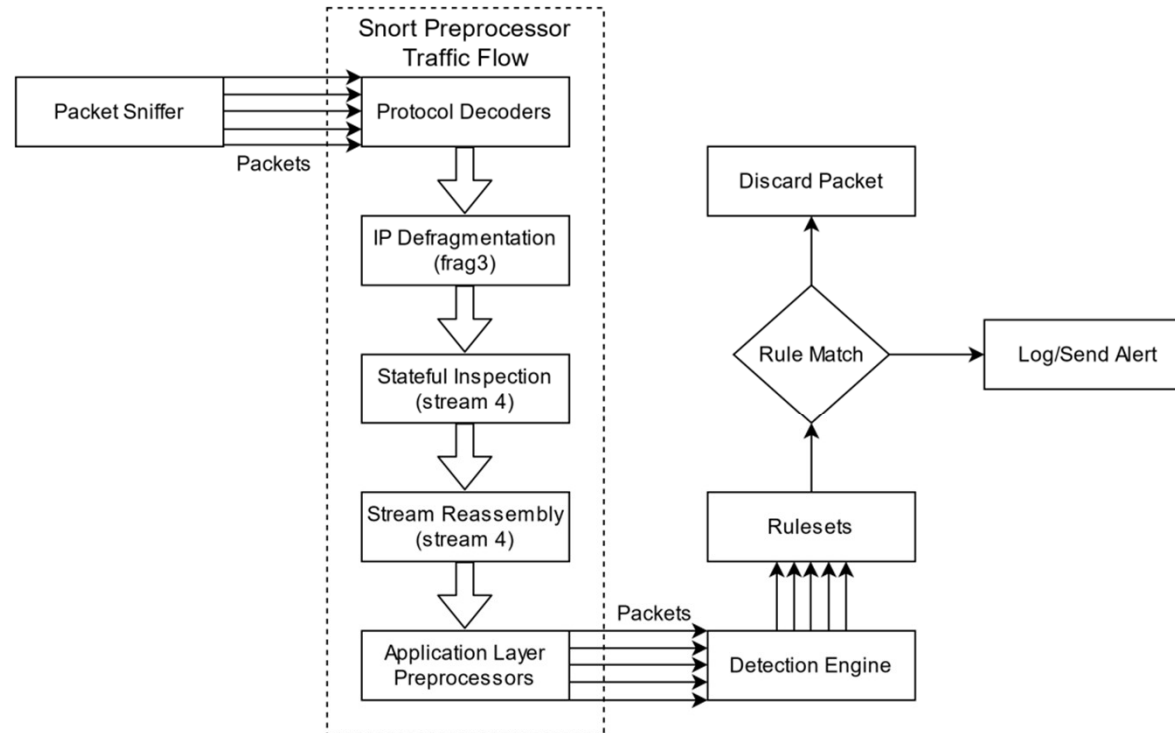
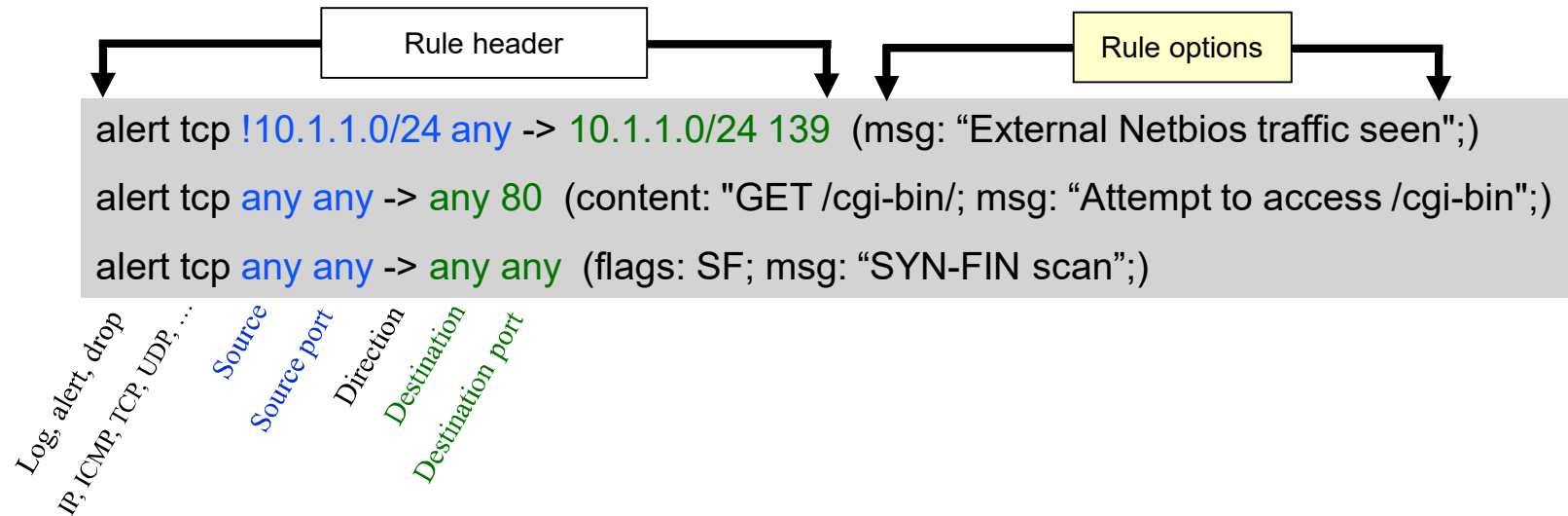


Figure 2.4: A simplified view of the Snort architecture and packet processing flow [20, pp. 40, 44, 227].

# Examples



*Snort subscriber ruleset:* Managed by CISCO

*Snort community ruleset:* Open source, daily updates, a subset of subscriber ruleset

# IDS Rules [Cisco IOS Firewall]

- **1100 IP Fragment Attack**
  - Triggers when any IP datagram is received with the "more fragments" flag set to 1 or if there is an offset indicated in the offset field.
- **1101 Unknown IP Protocol**
  - Triggers when an IP datagram is received with the protocol field set to 101 or greater. These protocol types are undefined or reserved and should not be used.
- **1102 Impossible IP Packet**
  - This triggers when an IP packet arrives with source equal to destination address. This signature will catch the so-called Land Attack.
- **2154 Ping of Death Attack**
  - Triggers when an IP datagram is received with the protocol field in the IP header set to 1 (ICMP), the Last Fragment bit is set, and  $(IP\ offset * 8) + (IP\ data\ length) > 65535$
- **3040 TCP - no bits set in flags**
  - Triggers when a TCP packet is received with no bits set in the flags field.
- **3041 TCP - SYN and FIN bits set**
  - Triggers when a TCP packet is received with both the SYN and FIN bits set in the flag field.
- **3042 TCP - FIN bit with no ACK bit in flags**
  - Triggers when a TCP packet is received with the FIN bit set but with no ACK bit set in the flags field.
- **3050 Half-open SYN Attack/SYN Flood**
  - Triggers when multiple TCP sessions have been improperly initiated on any of several well-known service ports. Detection of this signature is currently limited to FTP, Telnet, HTTP, and e-mail servers (TCP ports 21, 23, 80, and 25 respectively).
- **3153 FTP Improper Address Specified**
  - Triggers if a port command is issued with an address that is not the same as the requesting host.
- **3154 FTP Improper Port Specified**
  - Triggers if a port command is issued with a data port specified that is less than 1024 or greater than 65535.
- **4050 UDP Bomb**
  - Triggers when the UDP length specified is less than the IP length specified.
- **4100 TFTP Passwd File**
  - Triggers on an attempt to access the passwd file (typically /etc/passwd) via TFTP.
- Many more...



# Evading firewalls and Network IDS (NIDS) systems

See Security in IP and in TCP, both papers by CPNI,  
especially chapter 4.1 in IP paper.

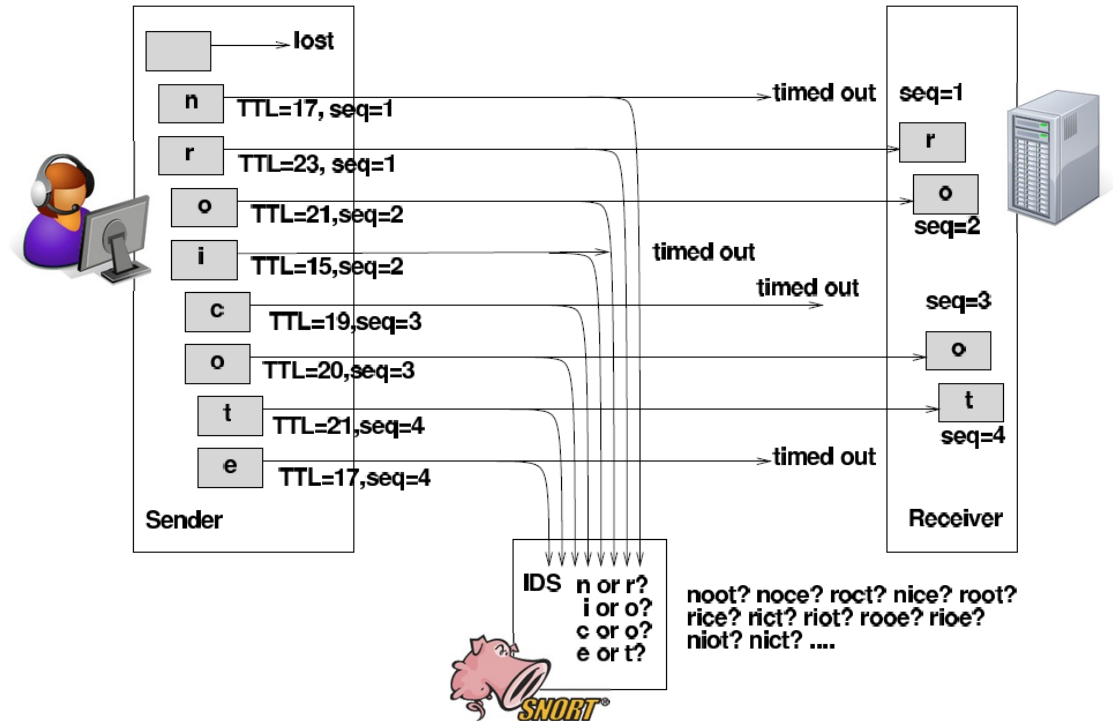
# Different techniques

- Main problem: NIDS must **know how end-systems interpret packets**
- Three main categories of problems
  - **Insert**: NIDS accepts packets end-systems reject
  - **Evade**: NIDS discards packets end-systems accept
  - **DoS**: NIDS system cannot keep up with the load
- Examples of packets that may or may not be accepted by hosts:
  - IPv4, TCP and UDP checksum error (no checksum in IPv6)
  - IP length != link layer length
  - TCP options present that may or may not be accepted
  - Expiring TTLs
  - IP fragmentation (different reassembly policies)
  - Overlapping TCP segments
- The analysis of application-level protocols can be extremely complex



# Expiring TTLs to evade IDS

- Can a firewall or IDS system analyze all combinations?
- Firewall can discard small TTL values that may time-out
- Or **normalize TTL values in border firewalls**
- RFC 5082: **Generalized TTL Security Mechanism**
  - Set incoming TTL to 255
  - Internal packets will have significantly lower numbers
  - Systems can distinguish source by looking at TTL



# The DF bit may cause packets to be lost

- The DF bit (Don't Fragment) may give an IDS system different info than a host depending on MTU
- Same packets may not reach IDS and target if fragmentation is needed

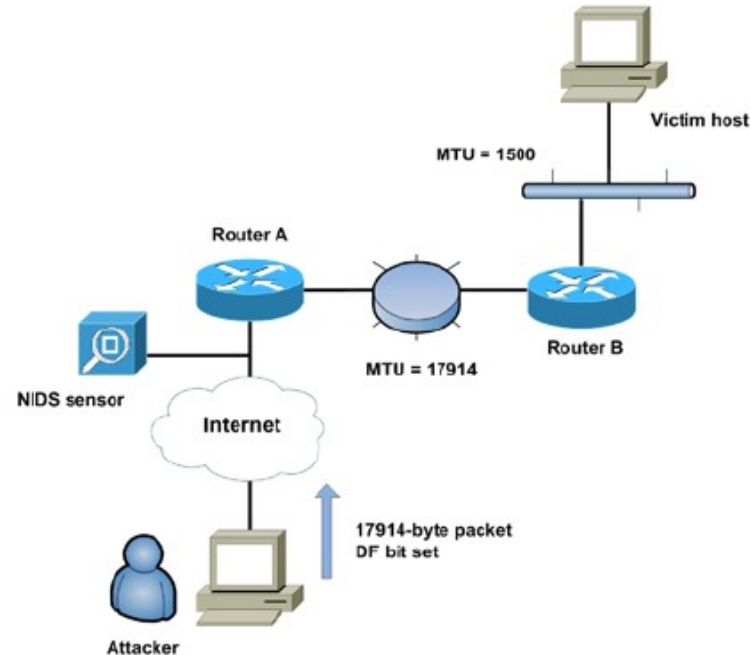
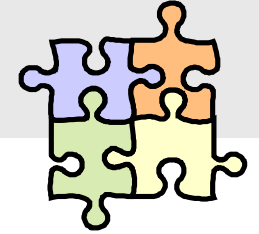


Figure 3: NIDS evasion by means of the Internet Protocol DF bit

Figure from "Security assessment of the IP protocol" by CPNI

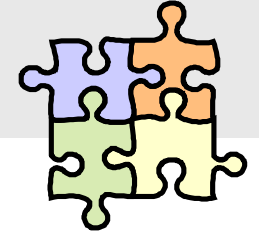


# IP Fragment Reassembly Timeout

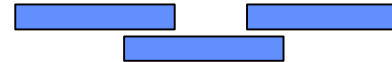


- Max time to wait for all fragments
  - All systems have a timer (RFC 792) – but waiting time differs
  - RFC 1122: 1-2 minutes, Linux 30 seconds
- If IDS system has shorter timeout than receiving host:
  - Attacker sends frag-1
  - IDS times out...
  - Attacker sends frag-2
  - Host reassembles datagram
  - IDS times out on frag-2
- If IDS has longer timeout (Snort default timeout 1 minute, Linux 30 seconds):
  - Attacker sends frag-1
  - Host times out...
  - Attacker sends frag-2
  - IDS reassembles datagram (looks ok)
  - Attacker resends different frag-1 (host still has frag-2, IDS may have dropped old fragments)
  - Host reassembles malicious datagram
  - IDS times out, drops frag-1

# Different IP reassembly policies



- Waiting consumes resources: DoS attack possible
  - Transmission of multiple last fragments (offset 64k) to allocate space
  - Defense: limit global buffer space allocated to fragments – should not affect regular traffic (Linux, BSD)
  - Some systems flush 50% of the fragment buffer when the threshold is reached
  - So what should the IDS system do?
- IP reassembly policies:
  - **First:** Always keep the first received fragment for each offset (never overwrite)
  - **Last:** Always keep the last received fragment (overwrite older) [RFC 791]
  - **Linux:** Left-trim i.e. keep fragments with lower offset
  - **BSD:** Right-trim
  - More possibilities exist... Poor IDS system!
- Snort: Available types are first, last, BSD, BSD-right, linux, windows and solaris. Default type is BSD
- In IPv6, routers do not fragment datagrams
  - Source must always send datagrams smaller than MTU but can fragment
  - Reassembly algorithm similar to IPv4
  - RFC 2460 contains rules: 60 second timeout for reassembly, check size <65,536, etc...



# IP Fragment Reassembly



- Two fully overlapping segments

AAAAAAAA

BBBBBBBB

CCCCCCCC

- Windows Vista and XP: prefer previous data (favor old)

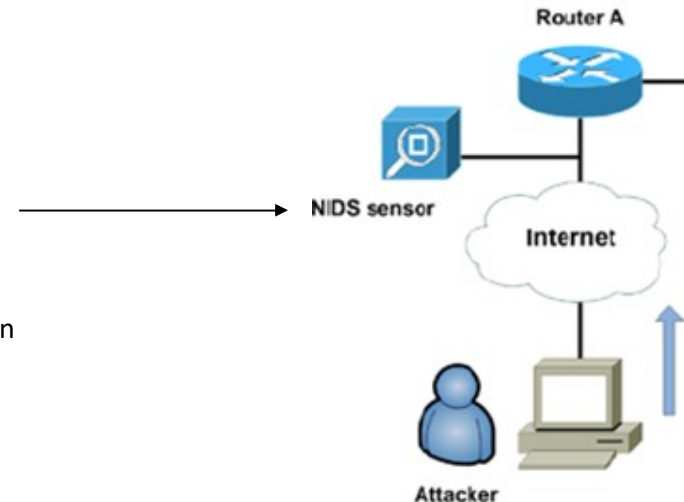
CCCCCCCCAAAAAAAA

- Linux: favor new

CCCCCCCCBBBBBBBB

# TCP sessions and IDS systems

- **“Handshake synchronization”**
  - Require SYN/ACK or full three-way handshake (TWH)
  - Good: impossible to force IDS to begin a session by sending only one faked segment
- **“Synchronize on data”**
  - IDS starts even if no TWH is seen
  - Advantage: Can pick up connections established before IDS was started
  - Disadvantage: Faked packets not part of any connection can trigger IDS to keep state
  - Ingress filtering needed
  - If new TWH seen, ignore old data?
- IDS system may not be able to see the direction of traffic on a network
  - Attacker may send a faked SYN + SYN/ACK + ACK to fool IDS system to start keeping state
  - Ingress filtering makes sure no one from the outside can fake traffic with an origin from the internal network
- Connection teardown
  - Always wait for FIN or RST?
  - Timeout?



# More issues with TCP



- NIDS must also understand ACKs, SACK and ICMP
  - Some hints to NIDS may be found in received ICMP messages
- TCP implementations may differ:
  - Accept data even if ACK flag not set?
  - Accept data in SYN segments?
  - Checksum errors in segments – some systems don't check
  - Accept or ignore options in non-SYN segments not negotiated earlier? Discard data too?
- In addition: TCP segments may also overlap
  - Not the same as IP fragmentation but problems are similar
  - TCP contains offset (seq. number) → overlaps possible
  - RFC 793: Trim to contain only new data – but not always followed (next slide)

# TCP Segment Reassembly



- Four overlapping segments:

**is\_is**  
**\_bad**  
**at\_m**  
**That**

Note: Not fragmentation, this is overlapping TCP segments

- Vista:

**This\_is\_bad**

- XP:

**That\_is\_bad**

- Linux:

**That\_is\_mad**

- Old data is always preferred over newer data (Vista)
- Behavior is novel
  - NIDSs will have to implement new strategy to prevent evasion attacks

# Summary – Evading IDS and FW

- Complex problem to deal with (lots of combinations)
  - Many ways to reassemble IP fragments (last, first, etc.)
  - Reassembly time-out, TTL, ...
  - TCP segments may also overlap, problems with following TCP state machines, etc.
- Firewalls and NIDS need to deal with IP fragmentation and overlaps
  - Analyze **all** reassembly combinations (opens for DoS attack?)
  - **Normalize TTL** values in border router
  - **Analyze datagrams differently depending on the target system (Snort)**
  - Consider discarding all fragments from the Internet (at least prevents some problems)
  - Always perform ingress filtering
- DoS attacks can be spawned against an NIDS
- IDS systems still need a lot of more research