Advanced Network Security

Firewalls and IDS

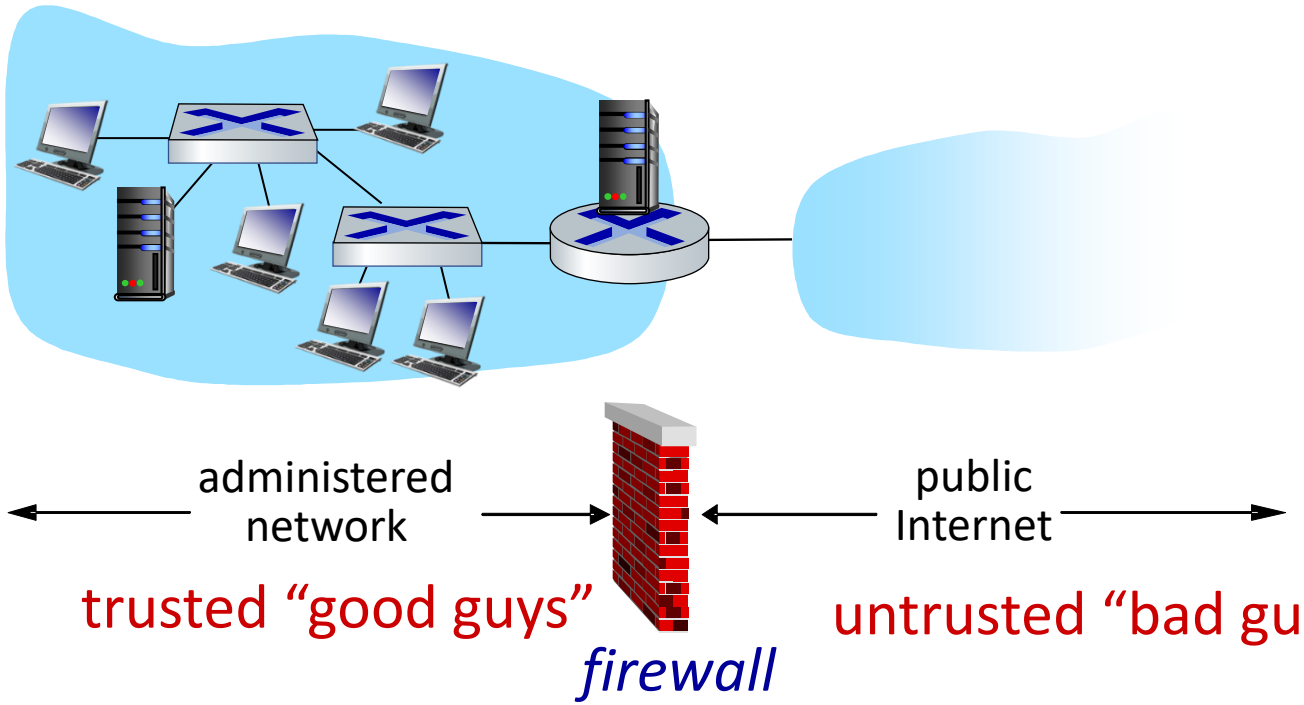
Amir Mahdi Sadeghzadeh, Ph.D.

Most slides have been adapted from Steven Bellovin, COMS W4180, Columbia University, 2006, and Mehdi Kharrazi, CE40-817, Sharif University of Technology, 2015

Firewalls

firewall

isolates organization's internal network from larger Internet, allowing some packets to pass, blocking others



Limitations of firewalls, gateways

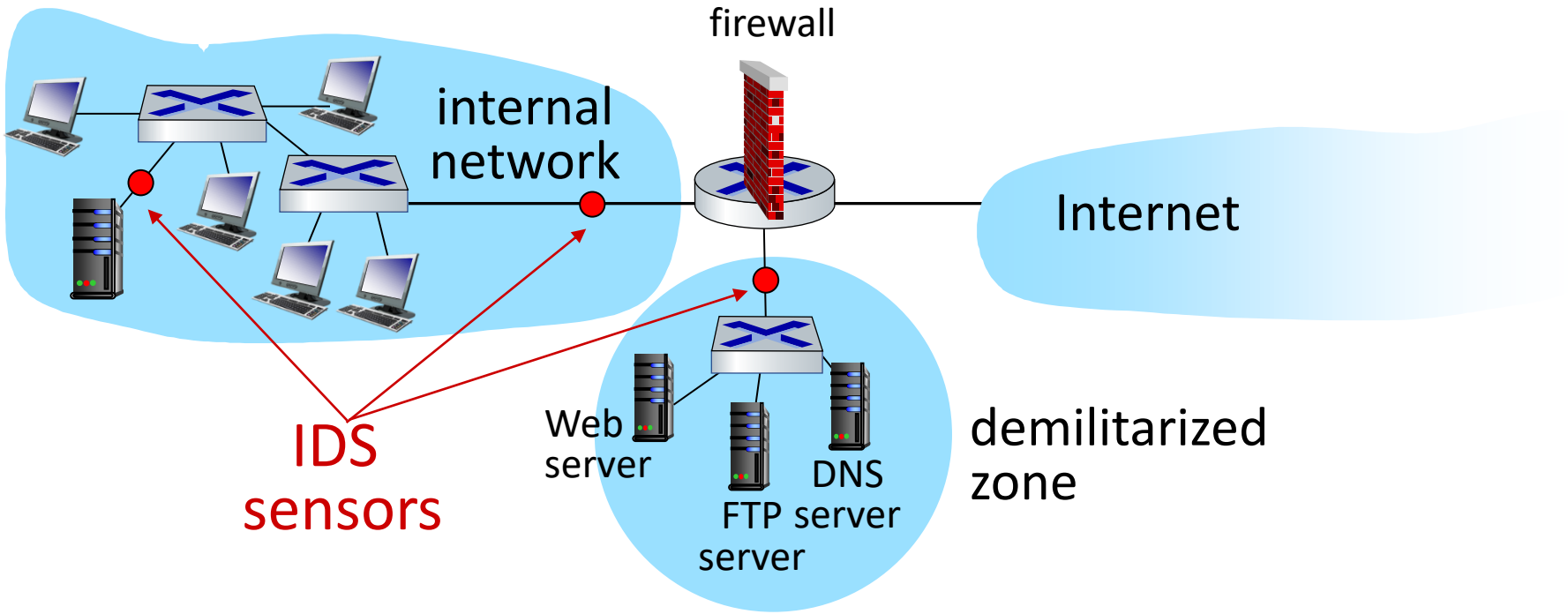
- Interfere with some networked applications
- Don’t solve many real problems
  - Buggy software (think buffer overflow exploits)
  - Bad protocol design (think WEP in 802.11b)
- Generally don’t prevent denial of service
- Don’t prevent insider attacks
- Increasing complexity and potential for misconfiguration

Intrusion detection systems

- packet filtering:
  - operates on TCP/IP headers only
  - no correlation check among sessions
- IDS: intrusion detection system
  - deep packet inspection: look at packet contents (e.g., check character strings in packet against database of known virus, attack strings)
  - examine correlation among multiple packets
    - port scanning
    - network mapping
    - DoS attack

Intrusion detection systems

multiple IDSs: different types of checking at different locations



Fighting intrusion

- Prevention: isolate from network, strict authentication measures, encryption
- Preemption:
  - “do to others before they do to you”
- Deterrence: dire warnings,
  - “we have a bomb too.”
    - Mutual Assured Destruction (MAD)
- Deflection: diversionary techniques to lure away
- Detection
- Counter attacks

Defense in Depth

- More generically, most single defenses can fail
- We always need defense in depth – multiple layers, of different designs and philosophies
- One such layer: Intrusion Detection Systems

What is IDS?

- An Intrusion Detection System (IDS) is a system that attempts to identify intrusions.
- Intrusion detection is the process of identifying and responding to malicious activity targeted at computing and networking resources.

- The goal of IDS is to detect fingerprints of malicious activity.

Examples of IDS in daily life

- Car Alarms
- House Alarms
- Surveillance Systems
- Spy Satellites, and spy planes

Elements of Intrusion Detection

- Primary assumptions:
  - System activities are observable
  - Normal and intrusive activities have distinct evidence
- Components of intrusion detection systems:
  - From an algorithmic perspective:
    - Features - capture intrusion evidence from audit data
    - Models - piece evidence together; infer attack
  - From a system architecture perspective:
    - Audit data processor, knowledge base, decision engine, alarm generation and responses

Where Are IDS Deployed?

- Host-based
  - Monitor activity on a single host
  - Advantage: better visibility into behavior of individual applications running on the host and network traffic
- Network-based (NIDS)
  - Often placed on a **router or firewall**
  - Monitor traffic, examine packet headers and payloads
  - Advantage: single NIDS can protect many hosts and look for global patterns

Host-Based IDSs

- Using OS auditing mechanisms
- BSM on Solaris: logs all direct or indirect events generated by a user
- Strace (the linux syscall tracer) for system calls made by a program
- Monitoring user activities
  - E.G., Analyze shell commands
- Monitoring execution of system programs
  - E.G., Analyze system calls made by sendmail

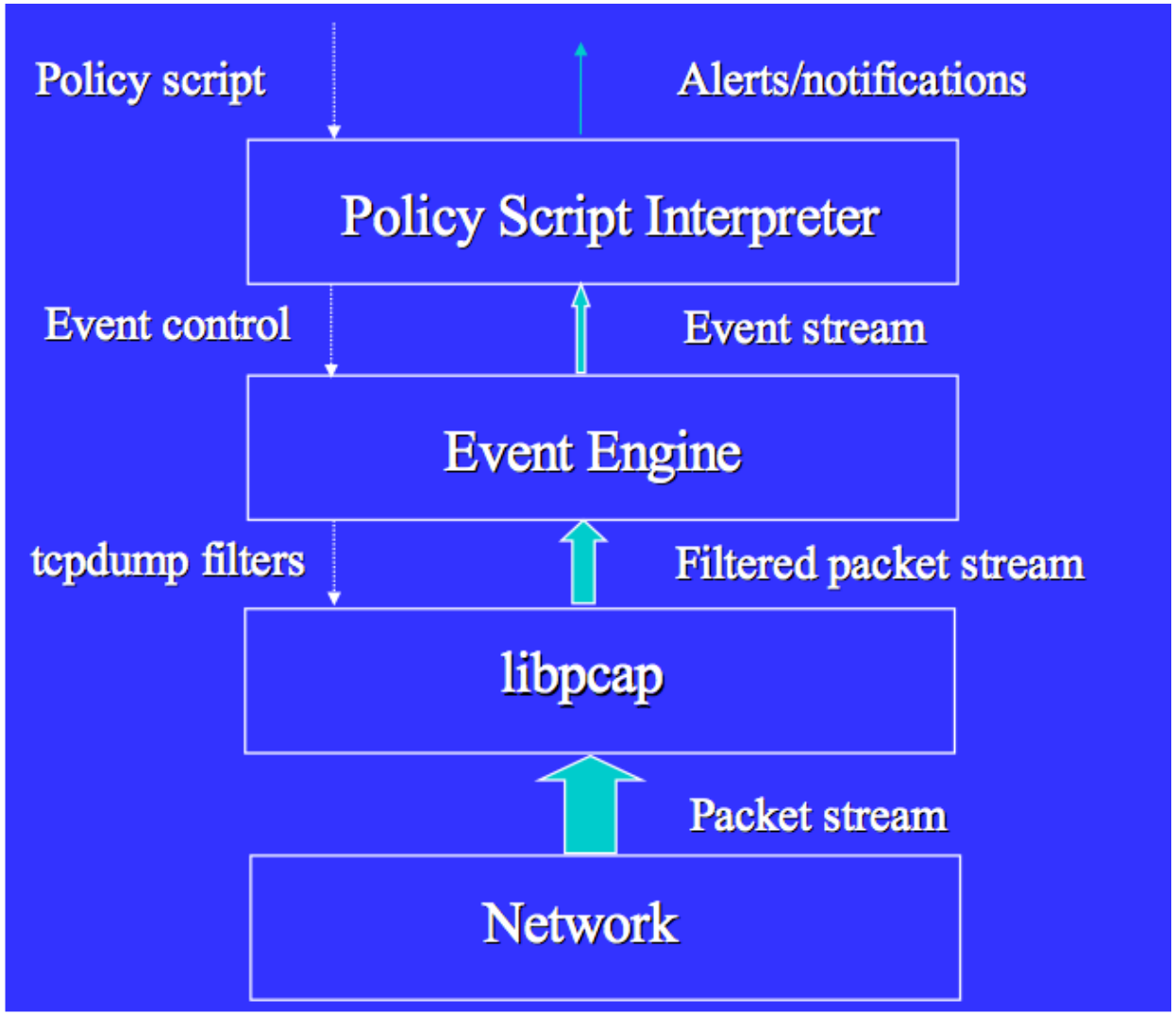
Basic Audit Modules (Hosts)

- eventLog - Uses the windows Event Logging system to track entries into all three of the windows event logs: System, Security, Application
- netstat - Uses the information from the program netstat to provide information about network usage on the machine
- health - Runs the program health to give current information about the system (CPU usage, mem usage, swap usage)
- ps - Uses information from the /proc virtual file system as a data source

Network IDSs

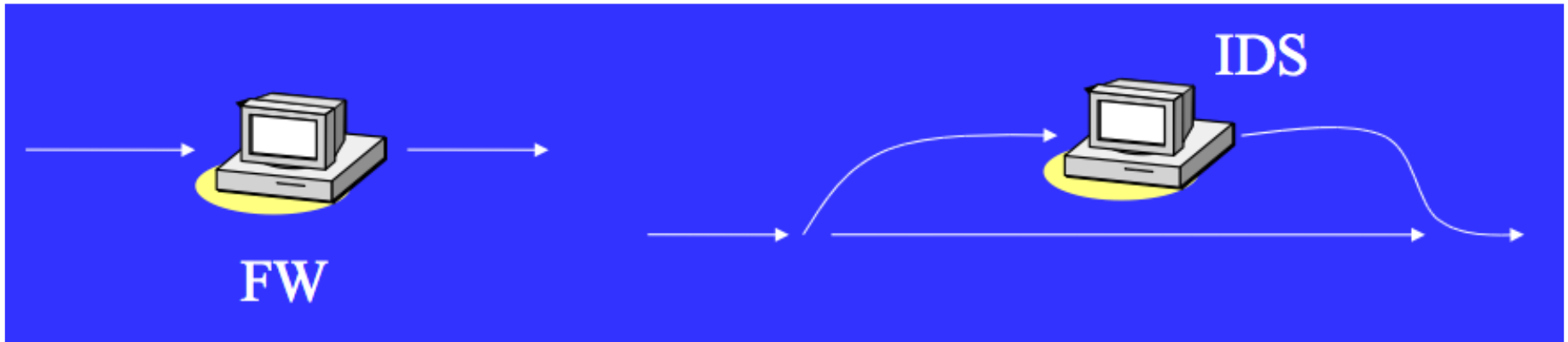
- Deploying sensors at strategic locations
  - E.G., Packet sniffing via tcpdump at routers
- Inspecting network traffic
  - Watch for violations of protocols and unusual connection patterns
- Monitoring user activities
  - Look into the data portions of the packets for malicious command sequences
- Problem
  - May be easily defeated by encryption
    - Data portions and some header information can be encrypted
  - Data volume

Architecture of Network IDS



Firewall Versus Network IDS

- Firewall
  - Active filtering
  - Fail-close
- Network IDS
  - Passive monitoring
  - Fail-open
- IPS (IDS + Firewall)



Requirements of Network IDS

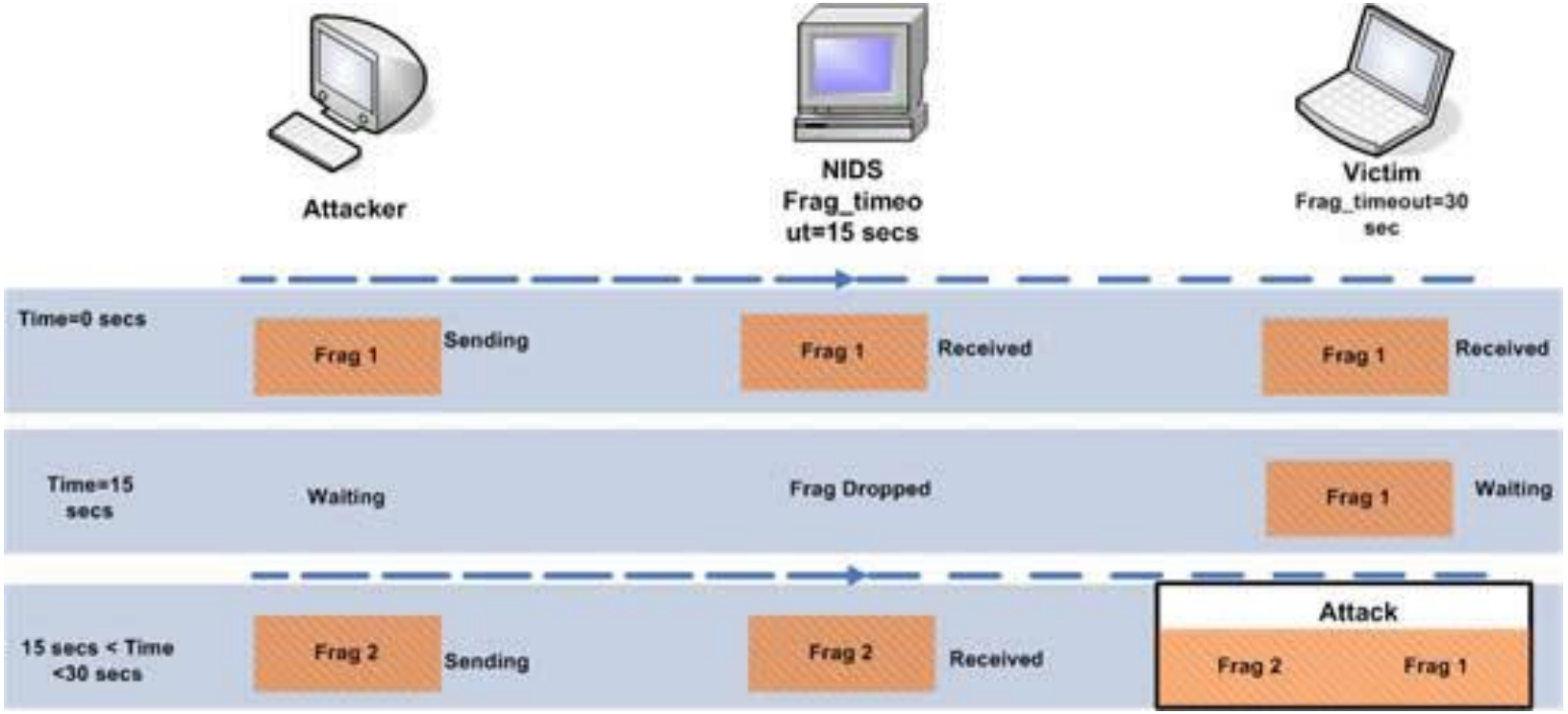
- High-speed, large volume monitoring
  - No packet filter drops
  - Why is it hard?
- Real-time notification
- Broad detection coverage
  - Precision, Recall, F-score
- Economy in resource usage
- Resilience to stress
- Resilience to attacks upon the IDS itself!

Eluding Network IDS

- What the IDS sees may not be what the end system gets.
  - Insertion and evasion attacks.
    - IDS needs to perform full reassembly of packets.
- But there are still ambiguities in protocols and operating systems:
  - E.G. TTL, fragments.
  - Need to “normalize” the packets.

Insertion Attack

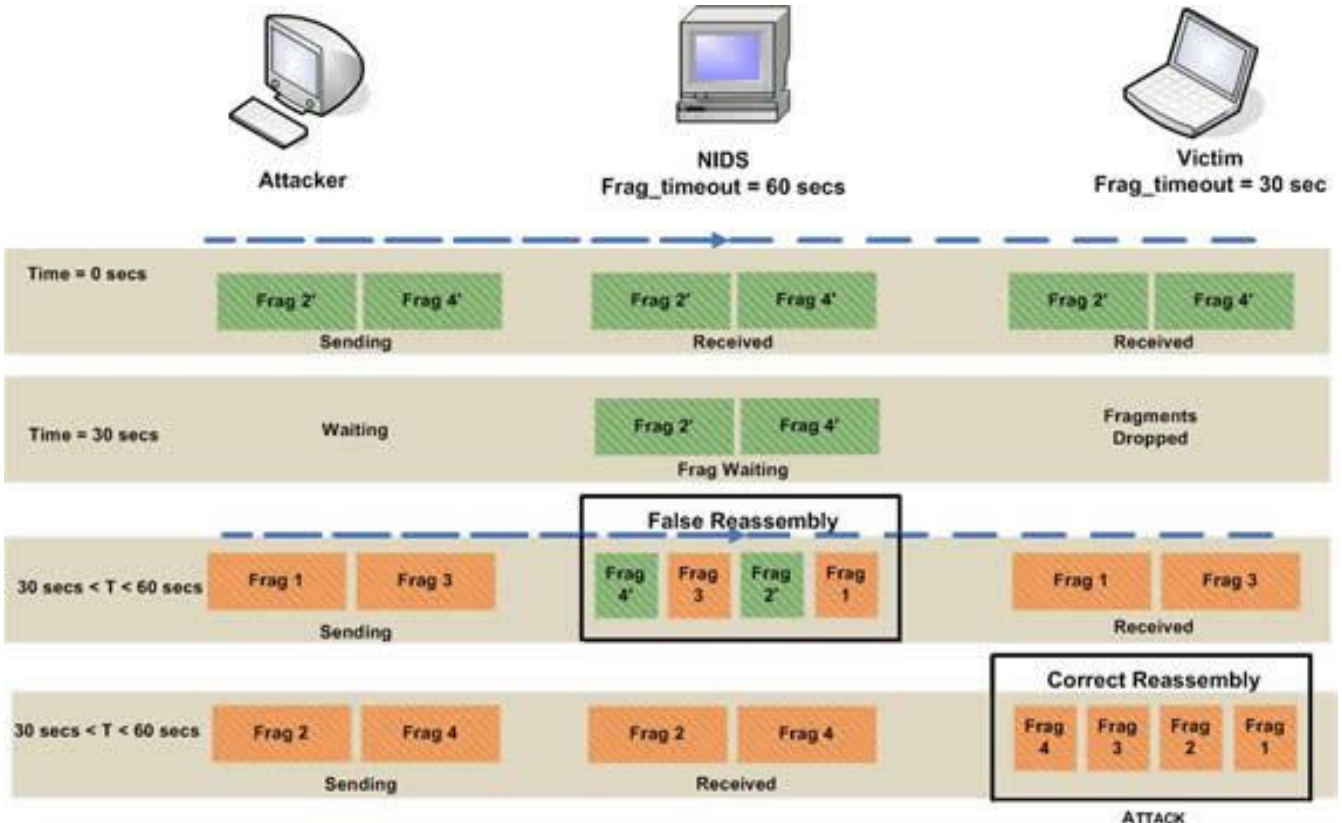
- IDS fragmentation reassembly timeout is 15 seconds
- The system is monitoring some Linux hosts which have a default fragmentation reassembly timeout of 30 seconds.



Source

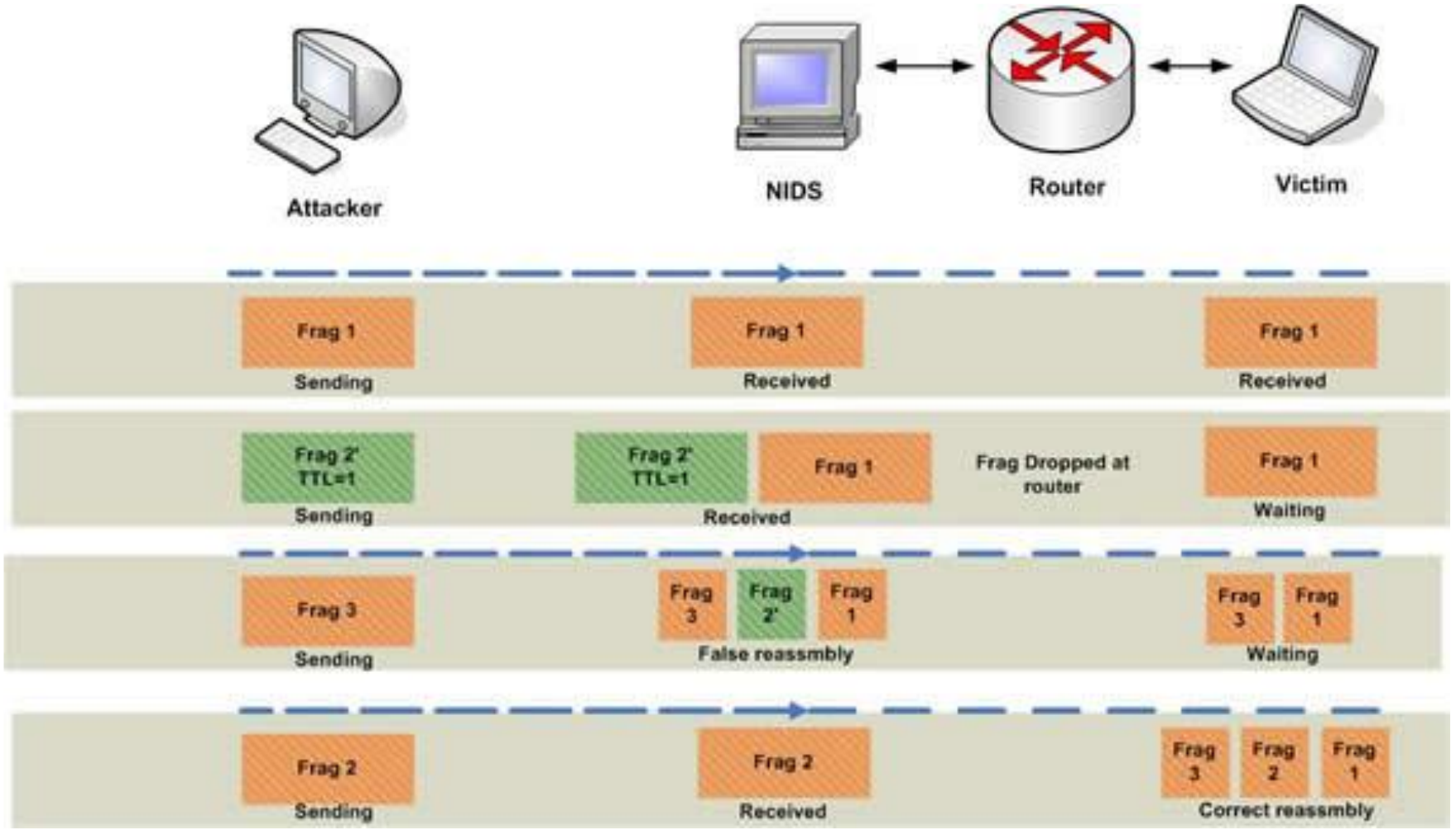
Insertion Attack

- By default, Snort has a fragment reassembly timeout of 60 seconds.
- Compare that to Linux/FreeBSD where it is 30 seconds.



Source

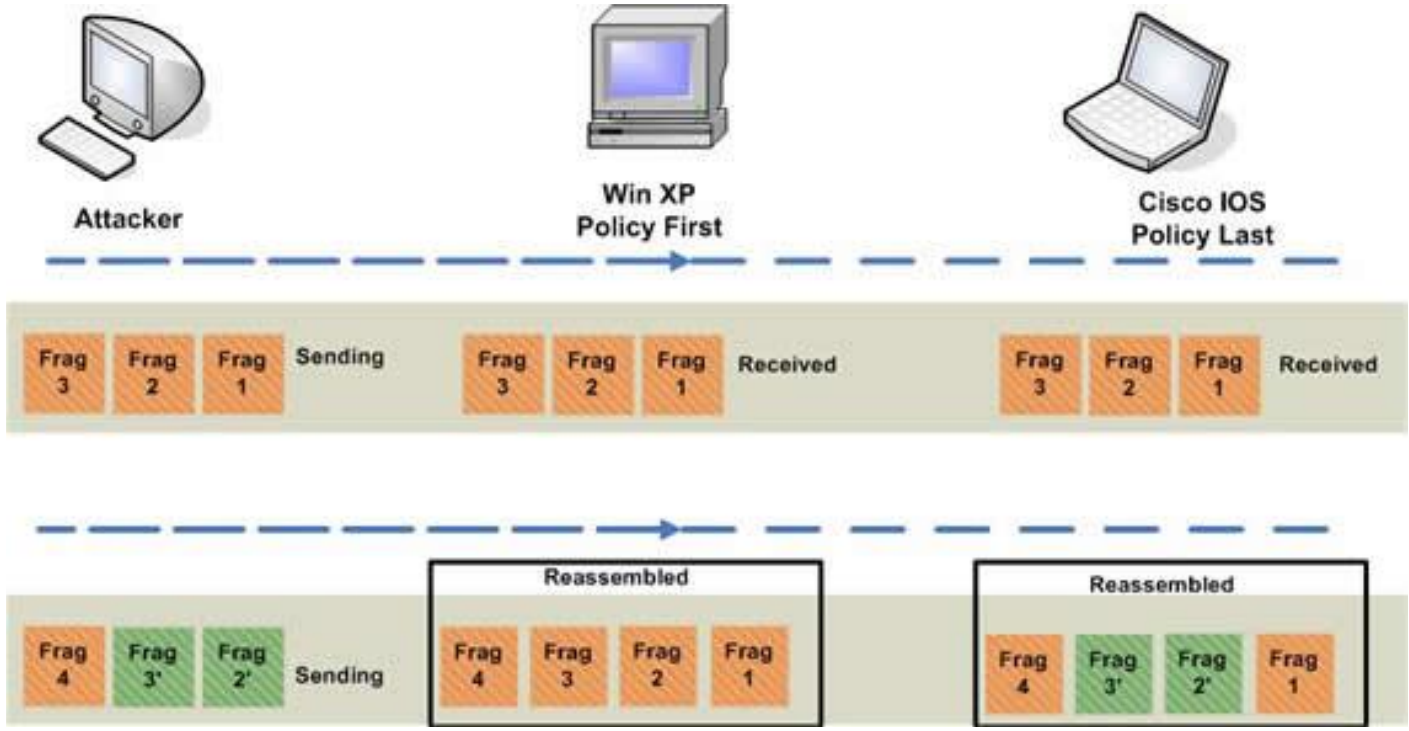
Insertion Attack



Source

Insertion Attack

- First
  - This is where the operating System favors the original fragments with a given offset. For example, Windows 95/98/NT4/ME/W2K/XP/2003.
- Last.
  - This is where the operating System favors the subsequent fragments with a given offset. For example, Cisco IOS.



Source

Snort evasion countermeasures

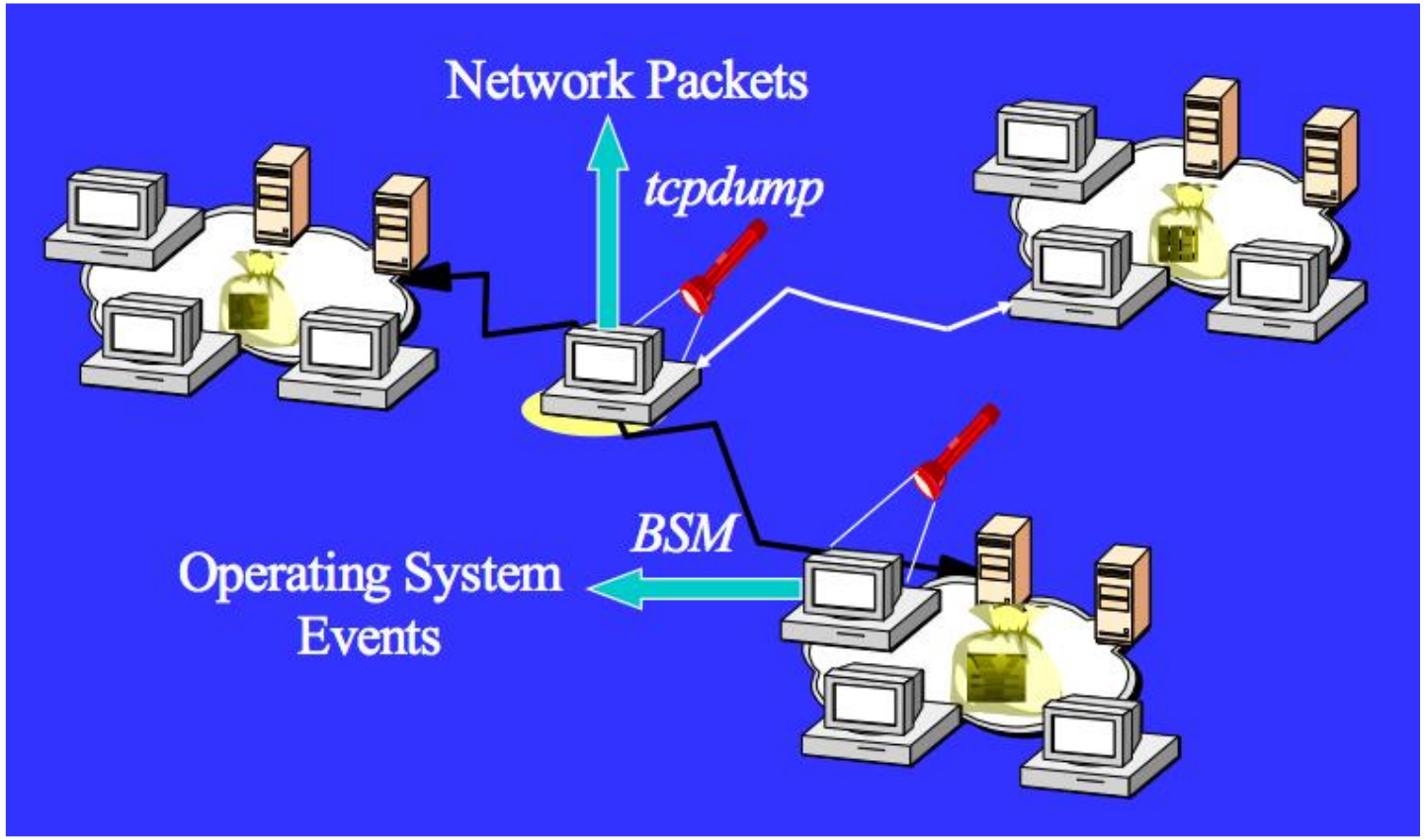
- Snort is the most popular NIDS.
  - The frag3 preprocessor is a target-based IP defragmentation module for Snort.
    - Allowing a user to identify the fragmentation reassembly method and the corresponding fragment timeout value that is applied to a particular destination IP address or subnet.
    - preprocessor frag3_engine: -policy bsd -bind_to 192.168.1.0/24 -timeout 30 -min_ttl 2

DoS Attacks on Network IDS

- Resource exhaustion
  - CPU resources
  - Memory
  - Network bandwidth
- Abusing reactive IDS
  - False positives

Hybrid NIDS and HIDS

- Sensors
  - Trust issue



Hybrid NIDS and HIDS

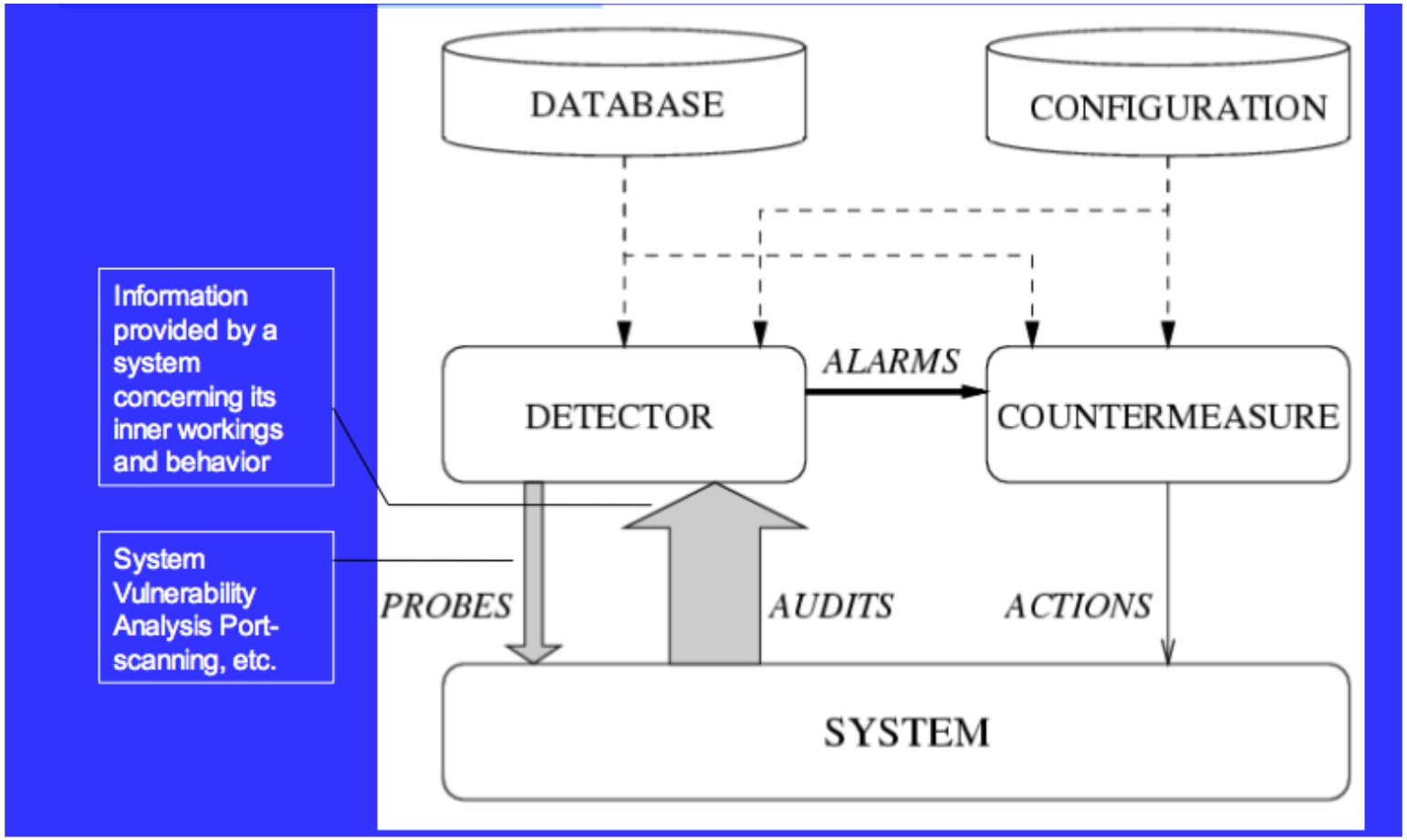
- Correlate information from multiple sources
- How do you trust your sources?

Taxonomy of IDS's

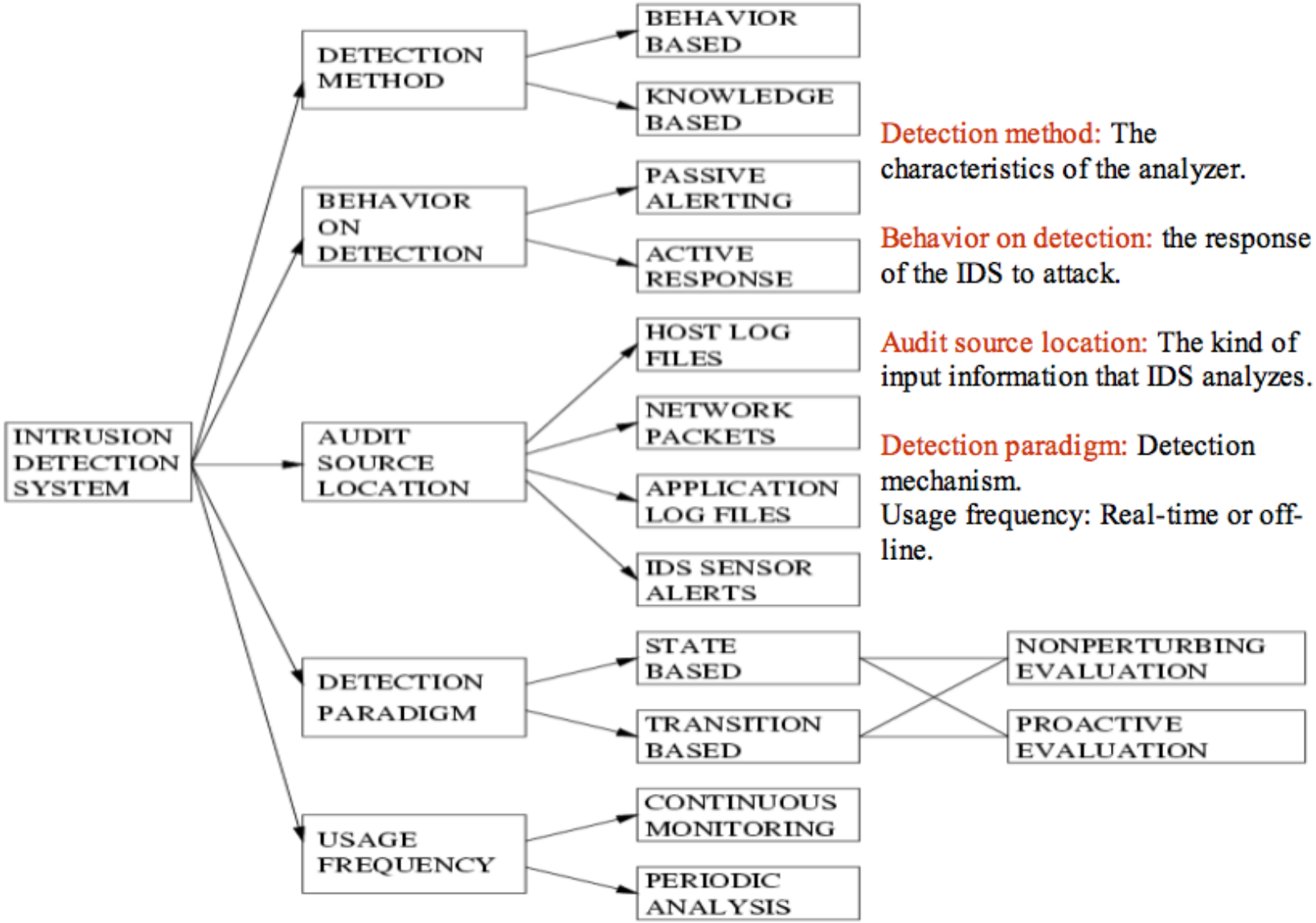
Intrusion Detection Approaches

- Modeling
  - Features: evidences extracted from audit data
  - Analysis approach: piecing the evidences together
    - Misuse detection (a.k.a. signature-based)
    - Anomaly detection (a.k.a. statistical-based)
- Deployment: Network-based or Host-based
- Development and maintenance
  - Hand-coding of "expert knowledge"
  - Learning-based on audit data

A Generic IDS



Characteristics of IDS



Detection method: The characteristics of the analyzer.

Behavior on detection: the response of the IDS to attack.

Audit source location: The kind of input information that IDS analyzes.

Detection paradigm: Detection mechanism.
Usage frequency: Real-time or off-line.

Detection Paradigm

- State-based versus transition-based IDS
  - State-based: Identifies intrusions on the states
  - Transition-based: Watches events that trigger transition from one state to another
- Non-perturbing versus pro-active analysis of state or transition
  - Non-perturbing: Acquire information transparently
  - Pro-active: Analysis by explicitly triggering events

IDS: Time aspect

- Real-time IDS
  - Analyzes the data while the sessions are in progress
  - Raises an alarm immediately when the attack is detected
- Off-line IDS
  - Analyzes the data after the information has been already collected
  - Useful for understanding the attackers' behavior

Knowledge-based IDS

- Good accuracy, bad completeness
  - Drawback
    - need regular update of knowledge
    - Difficulty of gathering the information
    - Maintenance of the knowledge is a time-consuming task
- Knowledge-based IDS
  - Misuse Detection
  - Specification-based Detection

Misuse Detection

- The system is equipped with a number of attack descriptions ("signature").
  - Then matched against the audit data to detect attacks.
- Signature
  - Sequences of system calls, patterns of network traffic, etc

- Pro: less false positives (But there still some!)
- Con: cannot detect novel attacks, need to update the signatures often.
- Approaches: pattern matching, security rule specification.

Misuse Detection (Signature-Based)

- Set of rules defining a behavioral signature likely to be associated with attack of a certain type
  - Example: buffer overflow
    - A setuid program spawns a shell with certain arguments
    - A network packet has lots of NOPs in it
    - A very long argument to a string function
  - Example: SYN flooding (denial of service)
    - Large number of SYN packets without ACKs coming back
- Attack signatures are usually very specific and may miss variants of known attacks
  - Why not make signatures more general?

Extracting Misuse Signatures

- Use invariant characteristics of known attacks
  - Bodies of known viruses and worms, port numbers of applications with known buffer overflows, RET addresses of stack overflow exploits
  - Hard to handle malware mutations
    - Metamorphic viruses: each copy has a different body
- Challenge: fast, automatic extraction of signatures of new attacks
- Honeypots are useful for signature extraction
  - Try to attract malicious activity, be an early target

Specification-based Detection

- Manually develop specifications that capture legitimate (not only previous seen) system behavior (all good states).
- Any deviation from it is an attack

- Pro: can avoid false-positive since the specification can capture all legitimate behavior.
- Con: hard to develop a complete and detailed specification, and error-prone.
- Approach: state machines

Today's IT Security Tools

- We make lists of bad behavior
  - Virus definitions
  - SPAM filters and blacklists
  - IDS signatures
  - Policies
- We distribute the lists to applications and detection systems
- They flag behavior that fits the pattern
- The system is about to collapse
  - Delays
  - Administrative Overhead
  - False positives

References

- Kurose, James F., and Keith W. Ross. "Computer networking: A top-down approach edition." Addision Wesley (2007), chapter 8.
- Steven Bellovin, COMS W4180, Columbia University, 2006
- Mehdi Kharrazi, CE40-817, Sharif University of Technology, 2015
- Vitaly Shmatikov, CS 361S, UT Austin, 2014