CSE508 Network Security

3/23/2016

Intrusion Detection

Michalis Polychronakis

Stony Brook University

Intrusion

"Any set of actions that attempt to compromise the integrity, confidentiality or availability of information resources" [Heady et al.]

"An attack that exploits a vulnerability which results to a compromise of the security policy of the system"
[Lindqvist and Jonsson]

Most intrusions…

- Are carried out remotely
- Exploit software vulnerabilities
- Result in arbitrary code execution or unauthorized data access on the compromised host
- ***Not the only way!***

Intrusion Method

Social engineering (phishing, spam, scareware, phone call, …)

Viruses/malware (~~disks, CD-ROMs~~, USB sticks, downloads, …)

Network traffic interception (access credentials, keys, tokens, …)

Password guessing (root:12345678, brute force cracking, …)

Physical access (reboot, keylogger, screwdriver, …)

Software vulnerability exploitation

…

Attack Source

Local

Unprivileged access ➔ privilege escalation

Physical access ➔ USB and other I/O ports, BIOS, wiretapping, memory/storage acquisition, bugging input devices, physical damage, …

Remote

Internet

Local network (Ethernet, WiFi, 3/4G, bluetooth, …)

Infected media (disks, CD-ROMs, USB sticks, …)

Phone (social engineering)

Less risk, more targets…

Attack Outcome

Arbitrary code execution

Privilege escalation

Disclosure of confidential information

Unauthorized access

DoS

Erroneous output

Destruction

…

Intrusion Detection

Intrusion detection systems monitor networks or hosts for malicious activities or policy violations

Detection (IDS): just generate alerts and log identified events

Prevention (IPS): in addition, react by blocking the detected activity

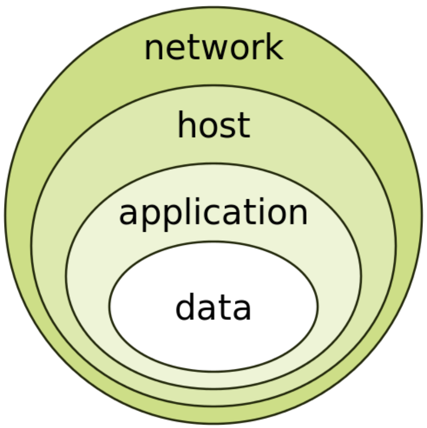
Defense in Depth

An IDS is not a silver bullet solution

- Just an additional layer of defense
- Complements existing protections, detectors, and policy enforcement mechanisms

There will always be new vulnerabilities, new exploitation techniques, and new adversaries

- Single defenses may fail
- Multiple and diverse defenses make the attacker's job harder



Defense in Depth

Securing systems retroactively is not always easy

- WiFi access points, routers, printers, IP phones, mobile phones, legacy devices, TVs, IoT, …

Detecting and blocking an attack might be easier/faster than understanding and fixing the vulnerability

- Immediate response vs. long-term treatment

Focus not only on detecting attacks

- But also on their side effects, and unexpected events in general
- Example: extrusion detection/data leak prevention ➔ detect data exfiltration

Situational Awareness

Understanding of what is happening on the network and in the IT environment

- Confirm security goals
- Identify and respond to unanticipated events

Diverse sources of data

- Passive/active network/host monitoring, scanning/probing, performance metrics/statistics, server/transaction logs, external (non IT) indicators, …
- Use data analytics to make sense of the increasing amount of data: identify features, derive models, observe patterns, …
- Data mining, machine learning, …

Basic Concepts: Location

An IDS can be a separate device or a software application

- Operates on captured *audit data*
- Off-line (e.g., periodic) vs. real-time processing

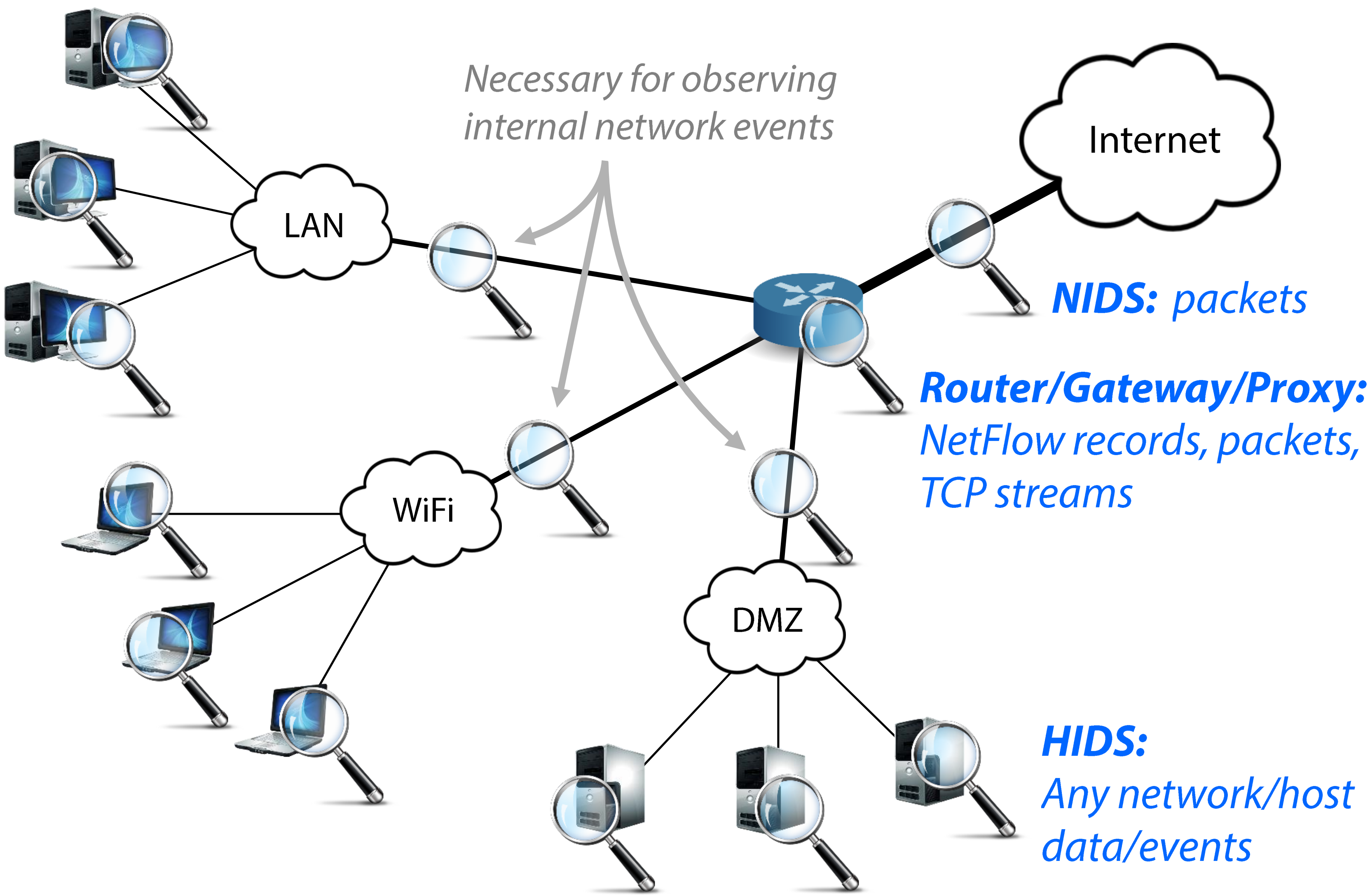
Network (NIDS)

- NetFlow records, raw packets, reassembled streams, …
- Passive (IDS) vs. in-line (IPS) operation
- Examples: Snort, Bro, Suricata, many commercial boxes, …

Host (HIDS)

- Login times, resource usage, user actions/commands, process/file/socket activity, application/system log files, registry changes, API calls, system calls, executed instructions, …
- Examples: OSSEC, El Jefe, AVs, registry/process/etc. monitors, network content scanners, …

Basic Concepts: Location



Deployment

NIDS: protect many hosts with a single detector

HIDS: install detector on each host (might not always be feasible)

Visibility

NIDS: can observe broader events and global patterns

HIDS: observes only local events that might not be visible at the network

Context

NIDS: packets, unencrypted streams (unless proxy-level SSL inspection)

HIDS: full picture

Overhead

NIDS: none (passive)

NIPS/Proxy: adds some latency

HIDS: eats up CPU/memory (overhead from negligible to complete hogging)

Subversion

NIDS: invisible in the network

NIPS/Proxy: failure may lead to unreachable network

HIDS: attacker may disable it and alter the logs (user vs. kernel level, in-VM vs. out-of-VM, remote audit logs)

Basic Concepts: Detection Method

Misuse detection

- Predefined patterns (known as "signatures" or "rules") of known attacks
- Rule set must be kept up to date
- Manual vs. automated signature specification (latter is *hard*)
- Can detect only *known* attacks, with adequate precision

Anomaly detection

- Rely on models of "normal" behavior
- Requires (re)training with an adequate amount of data
- Can detect previously unknown attacks
- Prone to false positives

IDS Challenges

Conflicting goals

- Zero-day attack detection
- Zero false positives

Resilience to evasion

Detection of targeted and stealthy attacks

Adaptability to a constantly evolving environment

- New threats, new topology, new services, new users, …
- Rule sets must be kept up to date according to new threats
- Models must be updated/retrained (*concept drift*)

Coping with an increasing amount of data

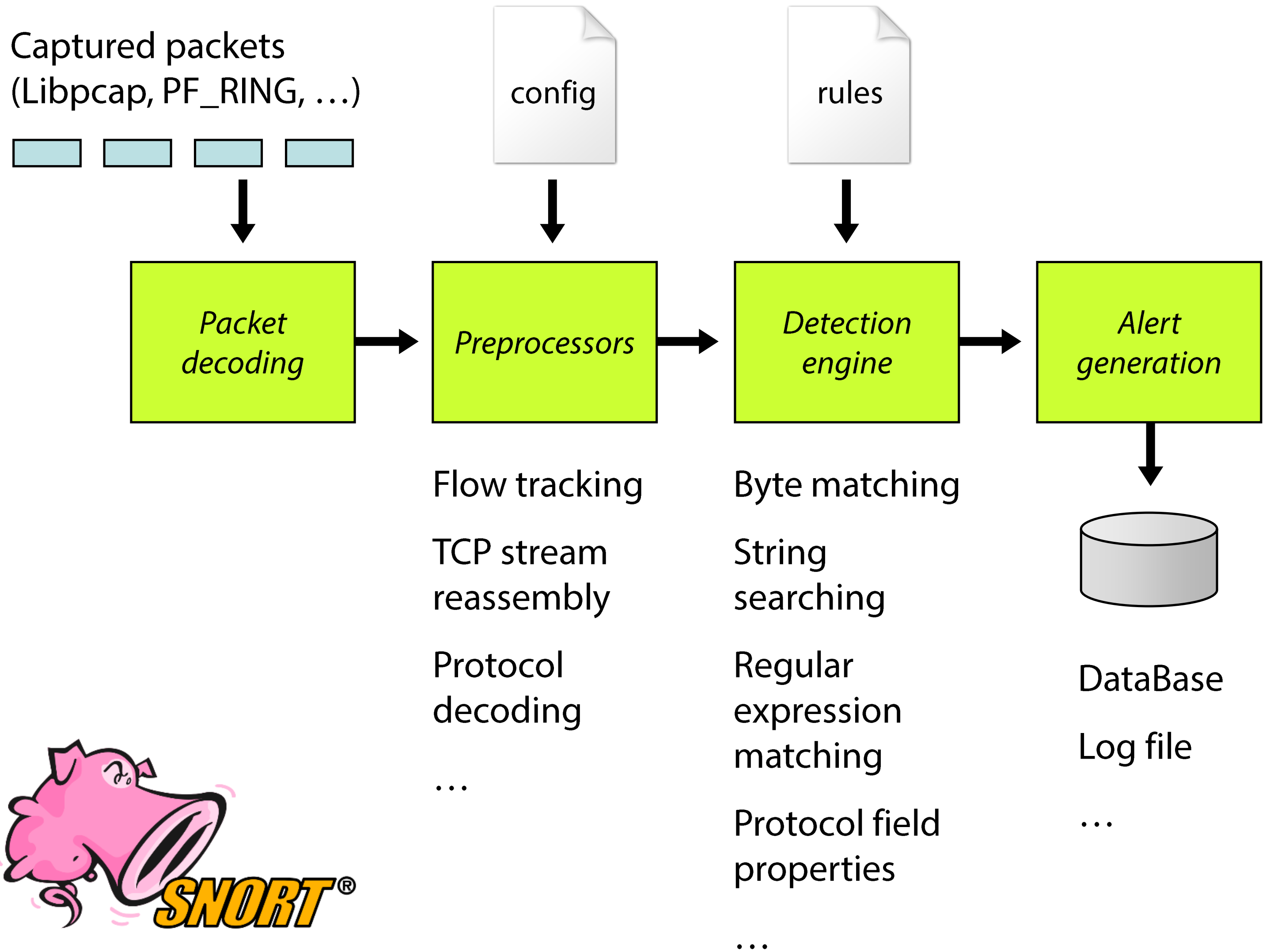
Popular Open-source Signature-based NIDS

Snort

Bro

Suricata

Use Case: Snort



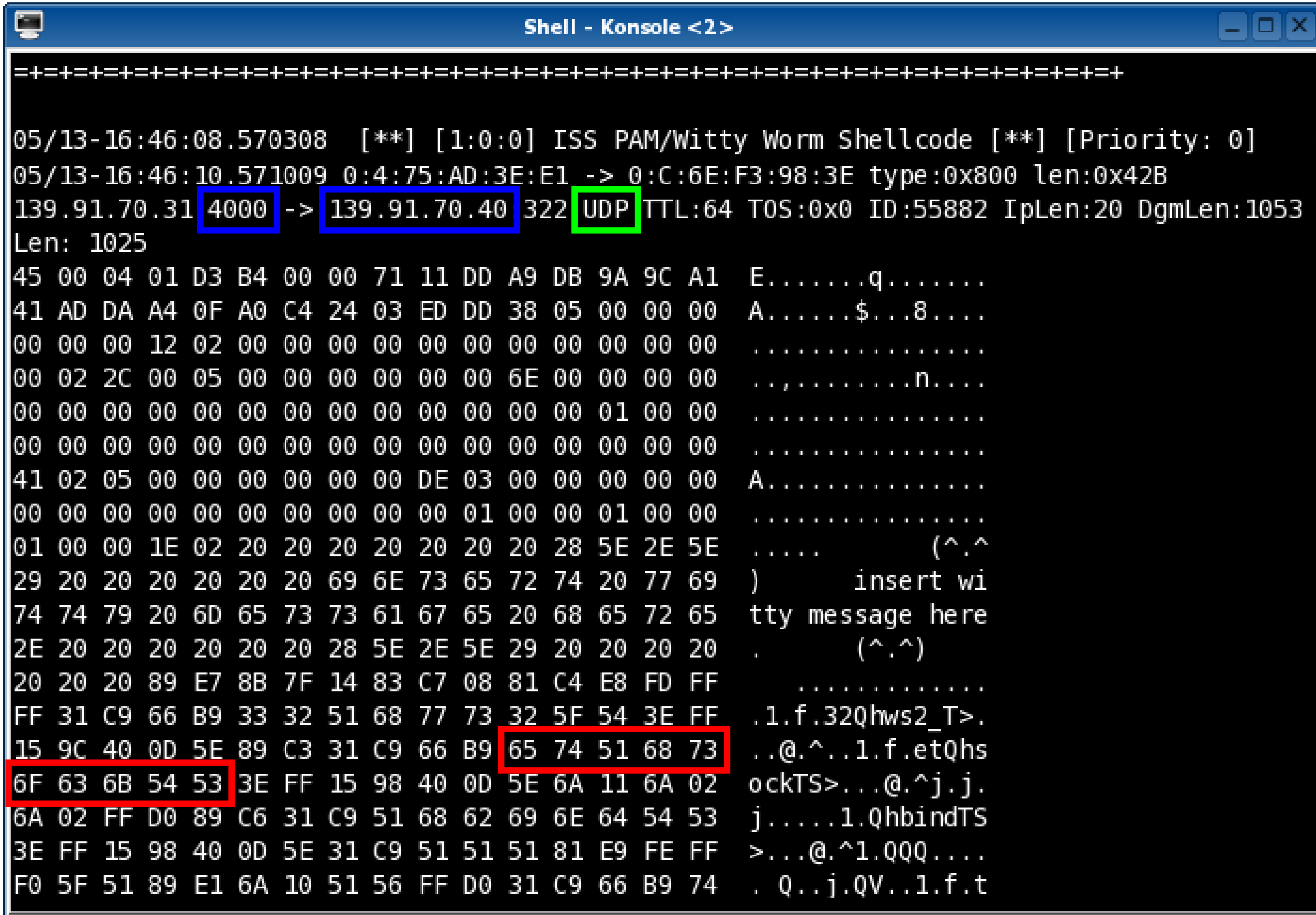
What is a Signature?

An attack description as seen at Layer 2-7

Witty worm Snort signature example:

action ***protocol*** ***source/destination*** ***content***

alert **udp** **any 4000 -> 193.92.123.0/24 any** (msg:"ISS PAM/Witty Worm Shellcode"; **content:"|65 74 51 68 73 6f 63 6b 54 53|";** **depth:246;** sid:1000078; rev:1;)


Shell - Konsole <2>
=+
05/13-16:46:08.570308 [**] [1:0:0] ISS PAM/Witty Worm Shellcode [**] [Priority: 0]
05/13-16:46:10.571009 0:4:75:AD:3E:E1 -> 0:C:6E:F3:98:3E type:0x800 len:0x42B
139.91.70.31 4000 -> 139.91.70.40 322 UDP TTL:64 TOS:0x0 ID:55882 IpLen:20 DgmLen:1053
Len: 1025
45 00 04 01 D3 B4 00 00 71 11 DD A9 DB 9A 9C A1 E.......q.......
41 AD DA A4 0F A0 C4 24 03 ED DD 38 05 00 00 00 A......$...8....
00 00 00 12 02 00 00 00 00 00 00 00 00 00 00 00 ....
00 02 2C 00 05 00 00 00 00 00 00 6E 00 00 00 00 ..,........n....
00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 ....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
41 02 05 00 00 00 00 00 00 DE 03 00 00 00 00 00 A...............
00 00 00 00 00 00 00 00 00 00 01 00 00 01 00 00 ....
01 00 00 1E 02 20 20 20 20 20 20 20 28 5E 2E 5E .....   (^.^
29 20 20 20 20 20 20 69 6E 73 65 72 74 20 77 69)   insert wi
74 74 79 20 6D 65 73 73 61 67 65 20 68 65 72 65 tty message here
2E 20 20 20 20 20 20 28 5E 2E 5E 29 20 20 20 20 .   (^.^)
20 20 20 89 E7 8B 7F 14 83 C7 08 81 C4 E8 FD FF   ....
FF 31 C9 66 B9 33 32 51 68 77 73 32 5F 54 3E FF .1.f.32Qhws2_T>.
15 9C 40 0D 5E 89 C3 31 C9 66 B9 65 74 51 68 73 ..@.^..1.f.etQhs
6F 63 6B 54 53 3E FF 15 98 40 0D 5E 6A 11 6A 02 ockTS>...@.^j.j.
6A 02 FF D0 89 C6 31 C9 51 68 62 69 6E 64 54 53 j.....1.QhbindTS
3E FF 15 98 40 0D 5E 31 C9 51 51 51 81 E9 FE FF >...@.^1.QQQ....
F0 5F 51 89 E1 6A 10 51 56 FF D0 31 C9 66 B9 74 . Q..j.QV..1.f.t

More Examples

String searching

```
alert ip $EXTERNAL_NET $SHELLCODE_PORTS -> $HOME_NET any
(msg:"SHELLCODE Linux shellcode"; content:"|90 90 90 E8 C0 FF
FF FF|/bin/sh"; classtype:shellcode-detect; sid:652; rev:9;)
```

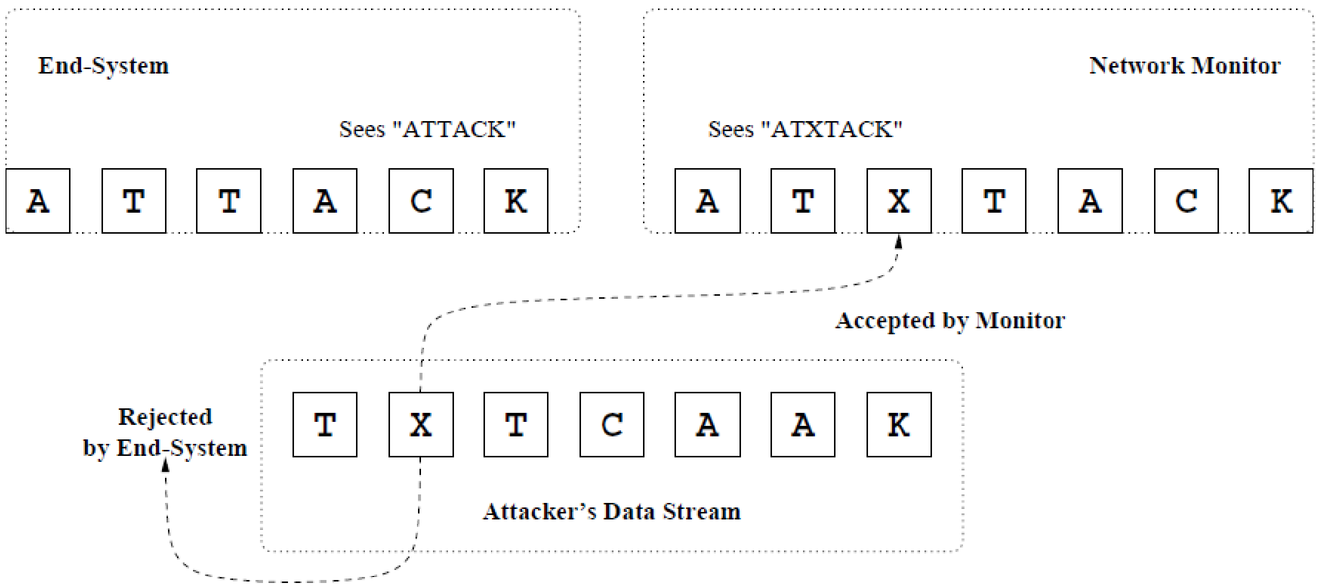
Strsearch + regexp matching + stateful inspection

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 10202:10203 (msg:"CA
license GCR overflow attempt"; flow:to_server,established;
content:"GCR NETWORK<"; depth:12; offset:3; nocase;
pcre:"/^\S{65}|\S+\s+\S{65}|\S+\s+\S+\s+\S{65}/Ri"; sid:3520;)
```

Stateful Inspection

Semantic gap: NIDS processes individual packets, while applications see a contiguous stream (TCP)

Potential for evasion

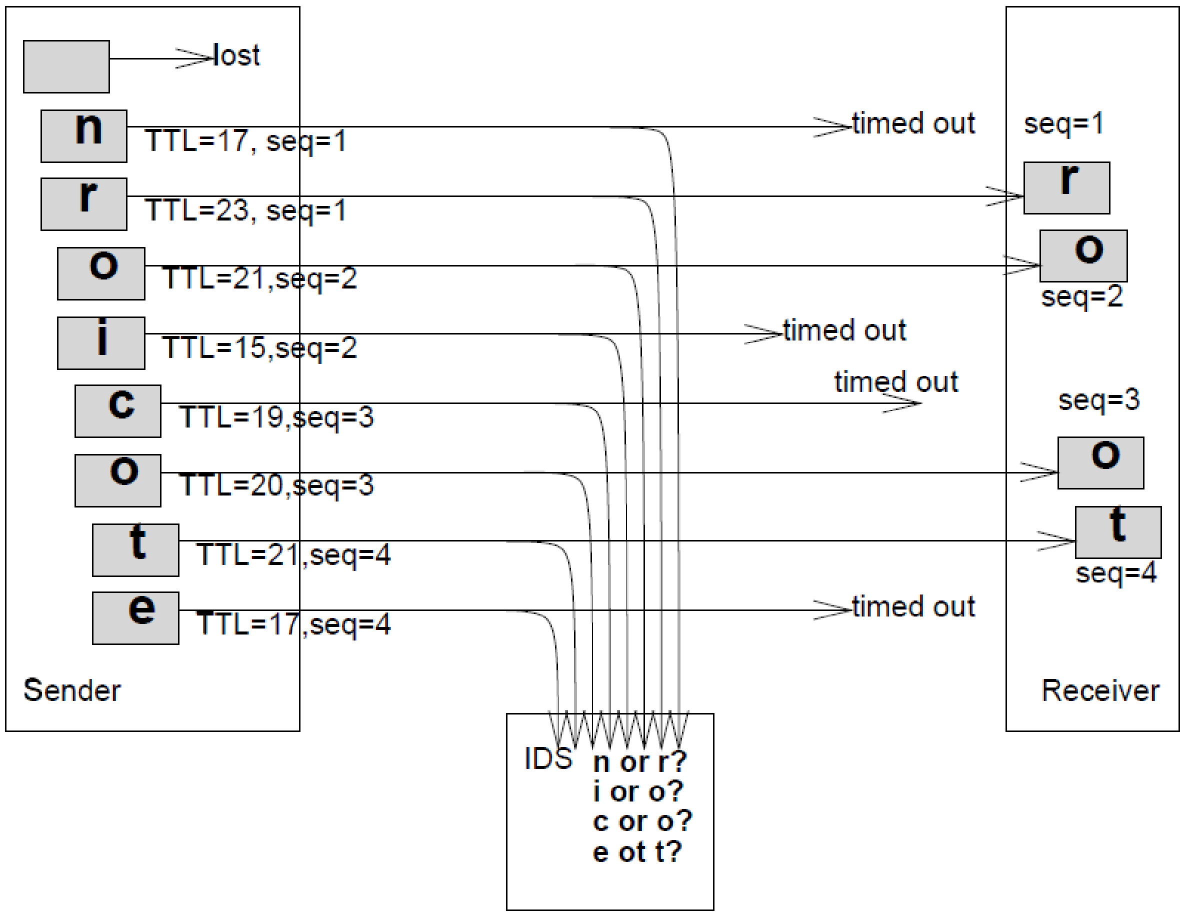


Solution: IP defragmentation, TCP stream reassembly

- Flow-level tracking: group packets into flows, track TCP state
- Stream reassembly: normalize and merge packets into streams

Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection – 1998

Different TCP stacks may treat corner cases differently…



Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics – USENIX Security 2001

Anomaly Detection

Training phase: build models of normal behavior

Detection phase: alert on deviations from the model

Many approaches

- Statistical methods, rule-based expert systems, clustering, state series modeling, artificial neural networks, support vector machines, outlier detection schemes, …

Good for noisy attacks

- Port scanning, failed login attempts, DoS, worms, …

Good for “stable” environments

- E.g., web server vs. user workstation

Anomaly Detection

Learning

Supervised

Labels available for both benign data and attacks

Semi-supervised

Labels available only for benign data

Unsupervised

No labels: assume that anomalies are very rare compared to benign events

Many possible features

Packet fields, payload content, connection properties, traffic flows, network metrics, system call sequences, code fragments, file attributes, statistics, …

Evaluating Intrusion Detection Systems

Accuracy is not a sufficient metric!

- Example: data set with 99.9% benign and 0.1% malicious events
- Dummy detector that marks everything as benign has 99.9% accuracy…

False positive: legitimate behavior was detected as malicious

False negative: an actual attack was not detected

Detection Result

Actual Event	Positive (alert)	Negative (silence)
Positive (malicious)	**TP**	**FN**
Negative (benign)	**FP**	**TN**

Precision = TP / (TP+FP)

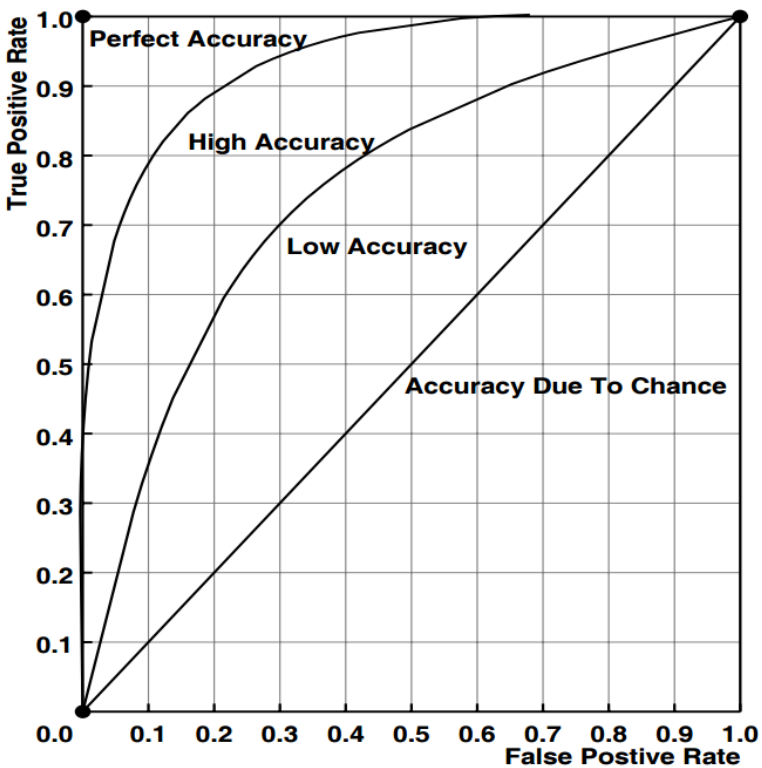
Recall (sensitivity) = TP / (TP + FN)

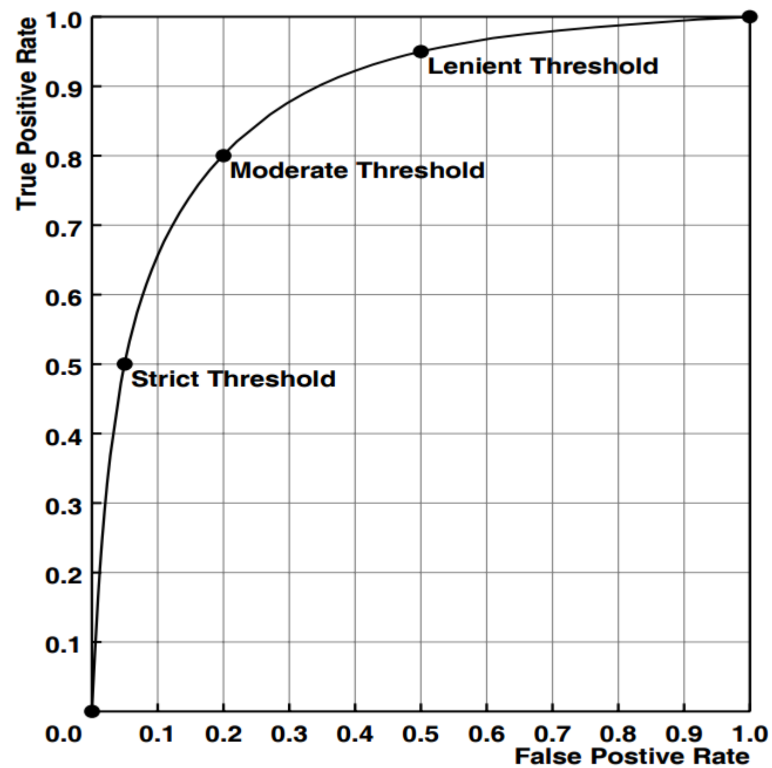
FP rate = FP / (FP + TN)

Receiver Operating Characteristic (ROC) Curve

Concise representation of a detector's accuracy

- Y axis: success rate of detecting signal events
- X axis: error rate of falsely identifying noise events





Proper Use of ROC Curves in Intrusion/Anomaly Detection - http://www.cs.newcastle.ac.uk/publications/trs/papers/871.pdf

Evasion – *"Stay under the radar"*

Both anomaly and misuse detection systems can be evaded by breaking the detector's assumptions

- Detectors rely on certain features
- Make those features look legitimate or at least non-suspicious

Many techniques

- Fragmentation
- Content mutation/polymorphism/metamorphism
- Mimicry
- Rate adjustment (slow and stealthy vs. fast and noisy)
- Distribution and coordination (e.g., DoS vs. DDoS)
- Spoofing and stepping stones
- …