



Chapter 15

Attack Detection

- ‰ Principles of Intrusion Detection Systems
- ‰ Categories
- ‰ Detection quality
- ‰ Tools

Introduction



‰ Definition: Intrusion

“An Intrusion is unauthorized access to and/or activity in an information system.”

□ Definition: Intrusion Detection

“The process of identifying that an intrusion has been attempted, is occurring or has occurred.”

National Security Telecommunications Advisory Committee (NSTAC) Intrusion Detection Subgroup

Introduction



‰ Intrusion Detection

- ‰ **Attack- / Invasion detection:** Tries to detect unauthorized access by outsiders
- ‰ **Misuse Detection:** Tries to detect misuse by insiders, e.g. users that try to access services on the internet by bypassing security directives
- ‰ **Anomaly Detection:** Tries to detect abnormal states within a network, e.g. sudden appearance of never used protocols, big amount of unsuccessful login attempts

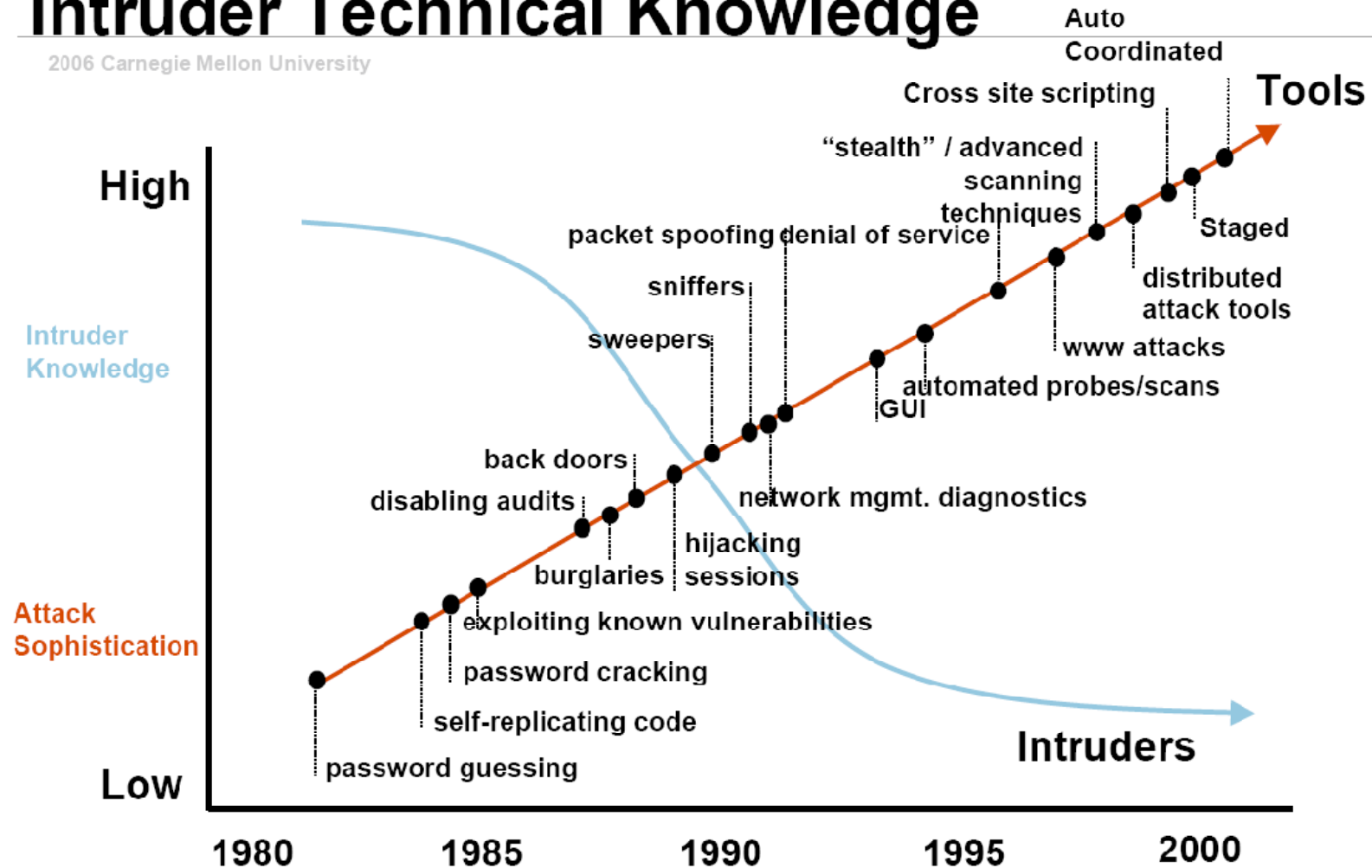
‰ Intrusion Prevention

- ‰ An IPS adds further functionality to an IDS. After detecting a possible attack the IPS tries to prevent the ongoing attack, e.g. by closing network connections or reconfiguring firewalls



Attack Sophistication vs. Intruder Technical Knowledge

2006 Carnegie Mellon University

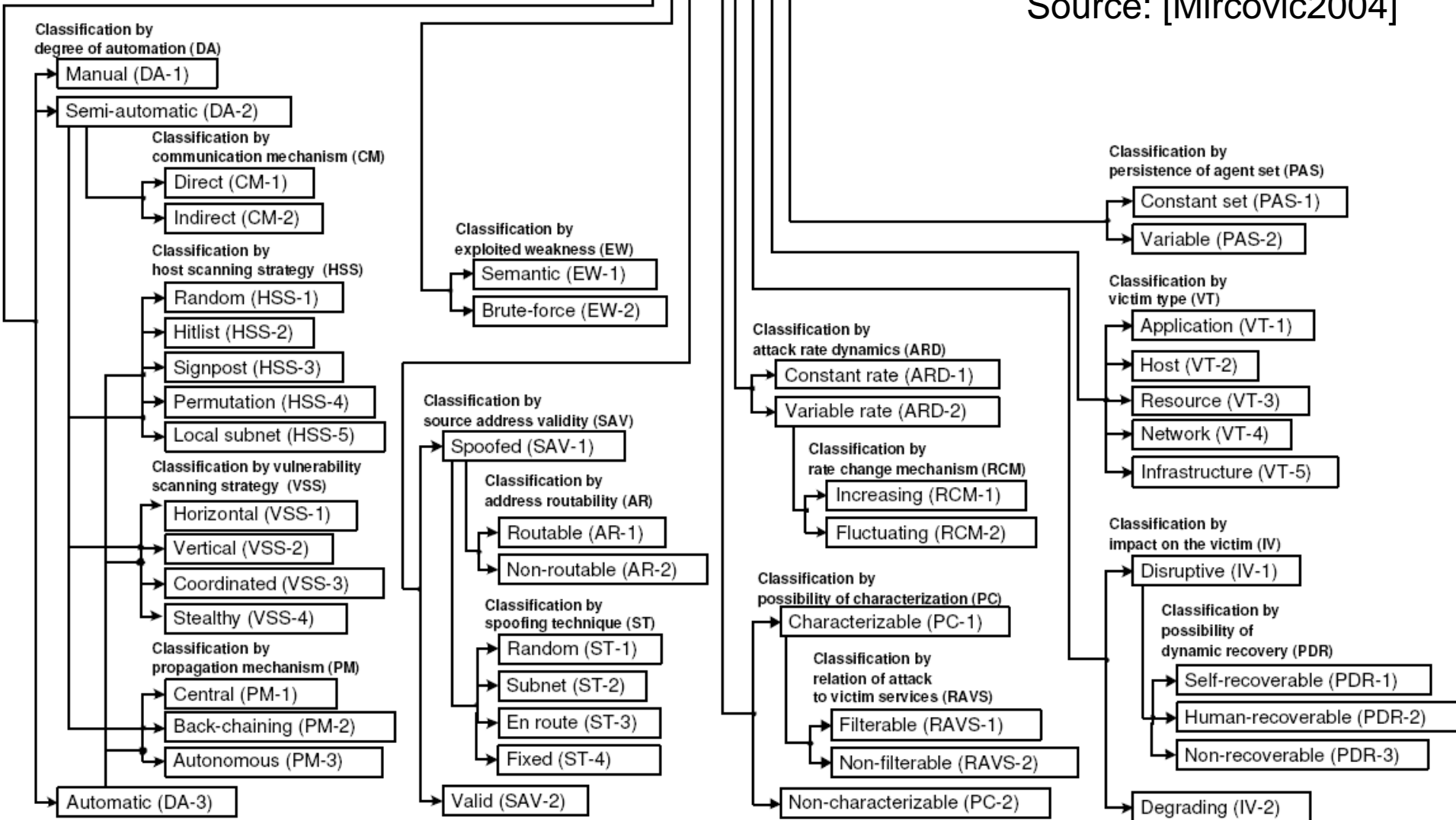


Attack Taxonomy



DDoS Attack Mechanisms

Source: [Mircovic2004]



Intrusion Detection



- ‰ Data collection issues

- ‰ Reliable and complete data

- ‰ Collection is expensive, collecting the right information is important

- ‰ Detection techniques

- ‰ **Signature-based** or knowledge-based detection

- ‰ **Anomaly detection**

- Response

- ‰ Counteracting an attack

- ‰ Evaluation

- ‰ System effectiveness, performance, network-wide analysis

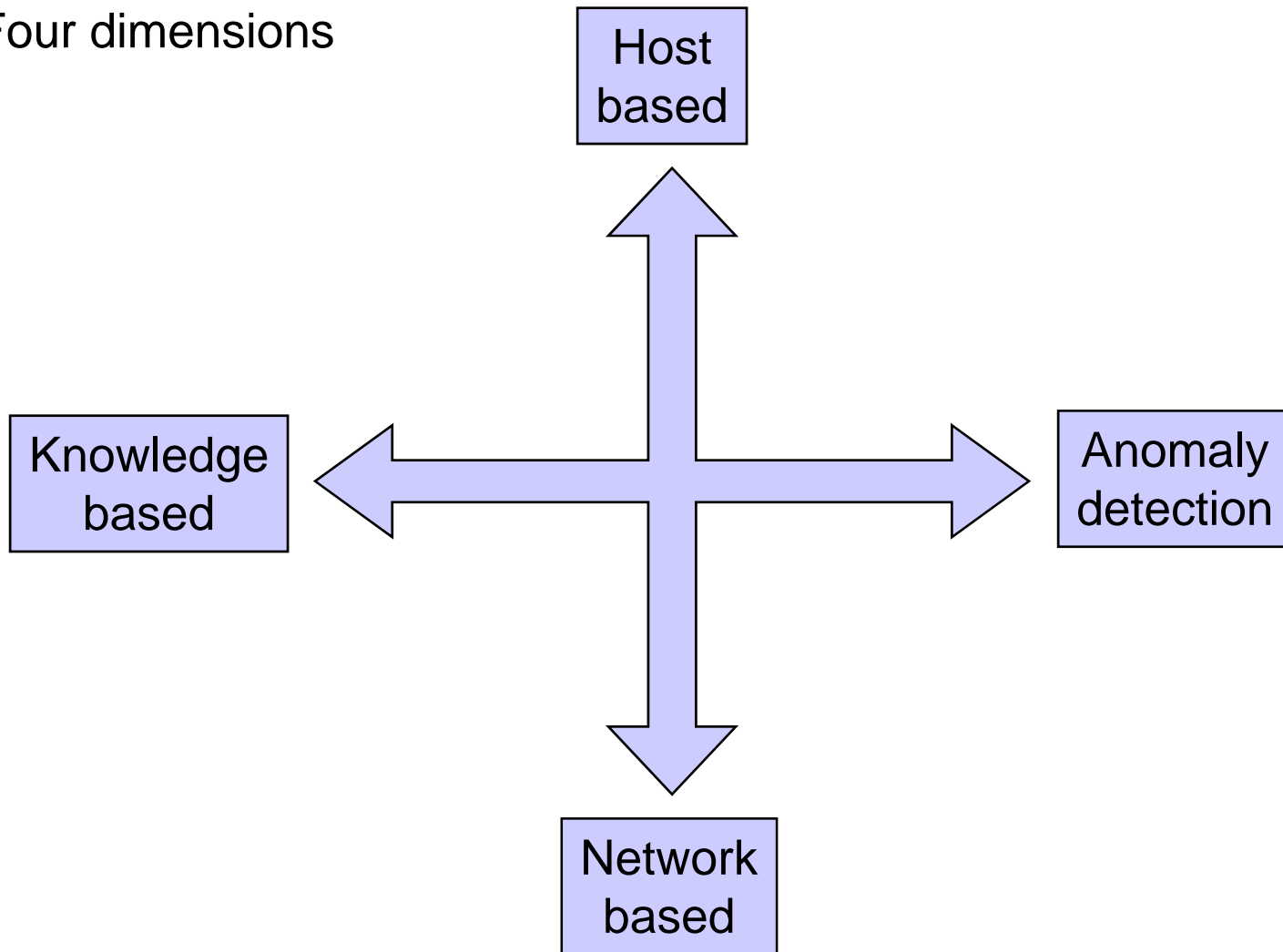
- ‰ False-positive rate

- ‰ False-negative rate

Classification of Attack Detection



Four dimensions



Classification of Attack Detection



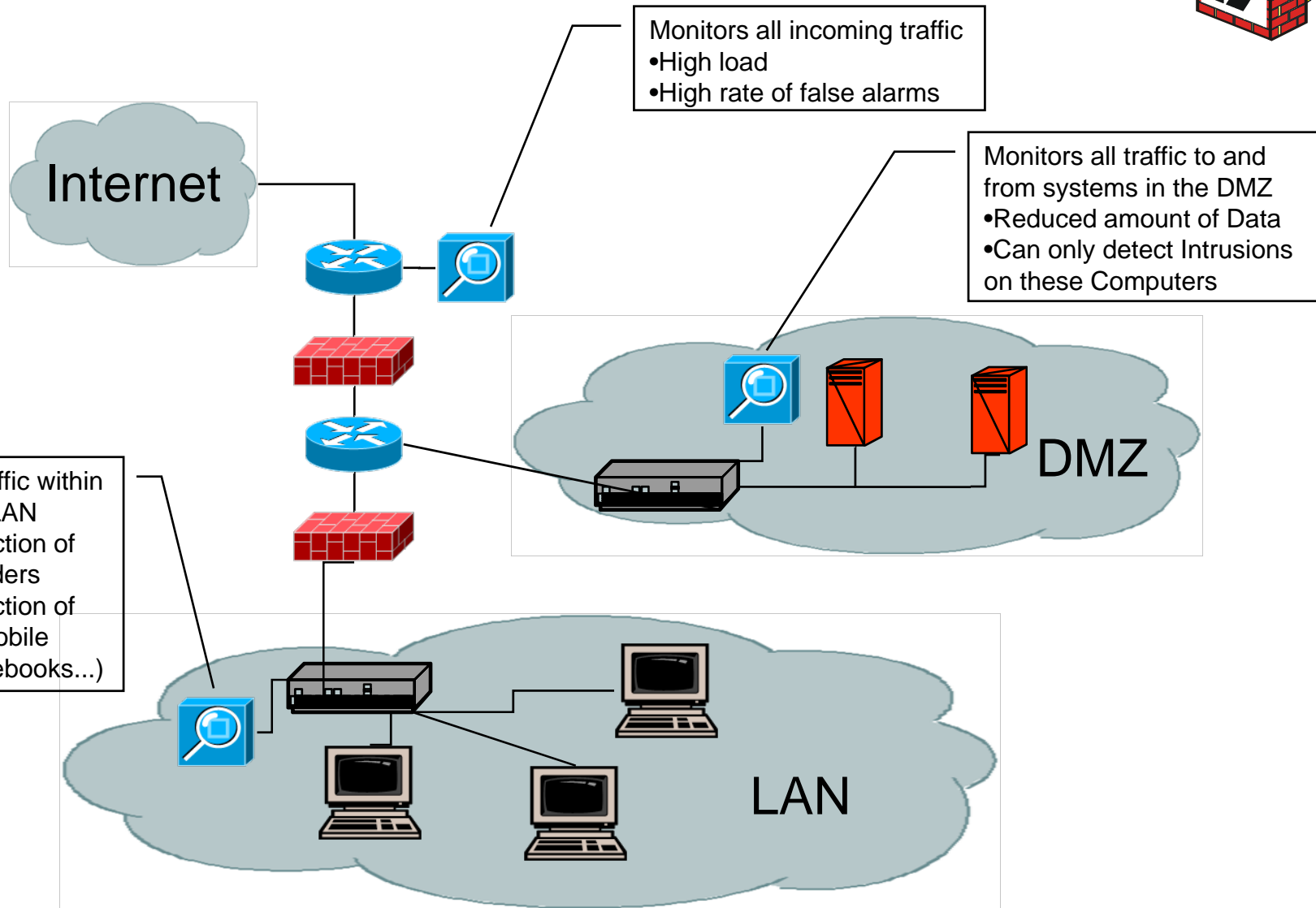
‰ Host Intrusion Detection Systems (HIDS)

- ‰ Works on information available on a system, e.g. OS-Logs, application-logs, timestamps
- ‰ Can easily detect attacks by insiders, as modification of files, illegal access to files, installation of Trojans or rootkits
- ‰ Problems: has to be installed on every System, produces lots of information, often no realtime-analysis but predefined time intervals, hard to manage a huge number of systems

‰ Network Intrusion Detection System (NIDS)

- ‰ Works on information provided by the network, mainly packets sniffed from the network layer. Uses signature detection (stateful), protocol decoding, statistical anomaly analysis, heuristical analysis
- ‰ Detects: DoS with buffer overflow attacks, invalid packets, attacks on application layer, DDoS, spoofing attacks, port scans
- ‰ Often used on network hubs, to monitor a segment of the network

Placement of a Network Intrusion Detection System



Knowledge-based Detection



- ‰ Based on signatures or patterns of well-known attacks

- ‰ Working principles

- ‰ Scan for attacks using well known vulnerabilities, e.g. patterns to attack IIS web server or MSSQL databases
- ‰ Scan for pre-defined numbers of ICMP, TCP SYN, etc. packets
- ‰ Patterns can be specified at each protocol level
 - „ Network protocol (e.g. IP, ICMP)
 - „ Transport protocol (e.g. TCP, UDP)
 - „ Application protocol (e.g. HTTP, SMTP)

- ‰ Pros

- ‰ Fast, requires few state information, low false-positive rate

- ‰ Cons

- ‰ Recognizes only known attacks

- ‰ Examples

- ‰ Snort, Bro



Signature-based Detection – Example: Snort

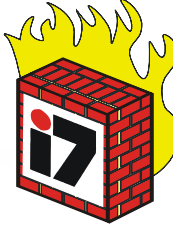


- ‰ Mainly signature based, each intrusion needs a predefined rule

```
alert tcp $HOME_NET any -> any 9996 \  
(msg:"Sasser ftp script to transfer up.exe"; \  
content:"|5F75702E657865|"; depth:250; flags:A+; classtype: misc-activity; \  
sid:1000000; rev:3)
```

- Three step processing of captured information (capturing is done by libpcap):
 - ‰ Preprocessing (normalized and reassembled packets)
 - ‰ Detection Engine works on the data and decides what action should be taken
 - ‰ Action is taken (log, alert, pass)

Anomaly Detection



- ‰ Based on the analysis of long-term and short-term traffic behavior

- ‰ Working principles

 - ‰ Scan for anomalies in

 - „ Traffic behavior

 - „ Protocol behavior

 - „ Application behavior

- ‰ Pros

 - ‰ Recognizes unknown attacks as well

- ‰ Cons

 - ‰ False-positive rate might be high

- ‰ Examples

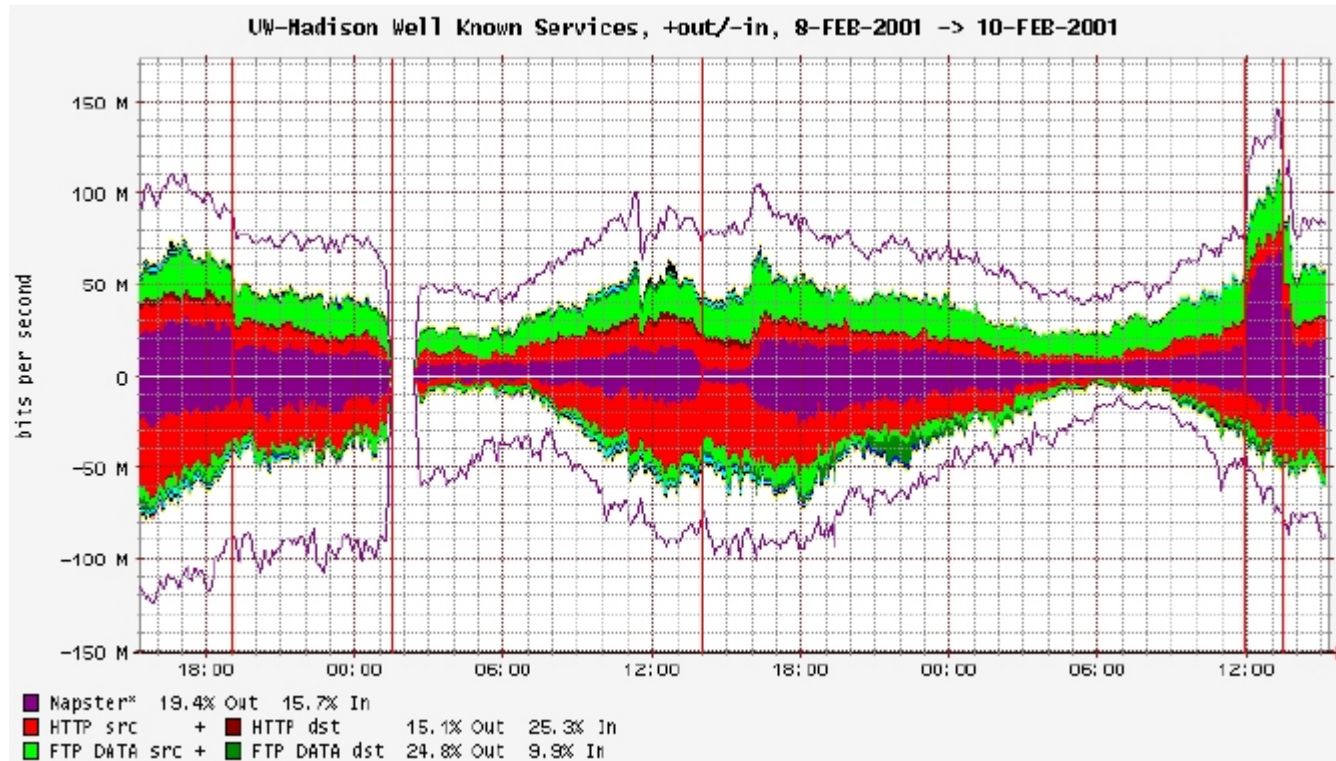
 - ‰ PHAD/ALAD, Emerald

Anomaly Identification



‰ Network operation anomalies

‰ Caused by configuration changes



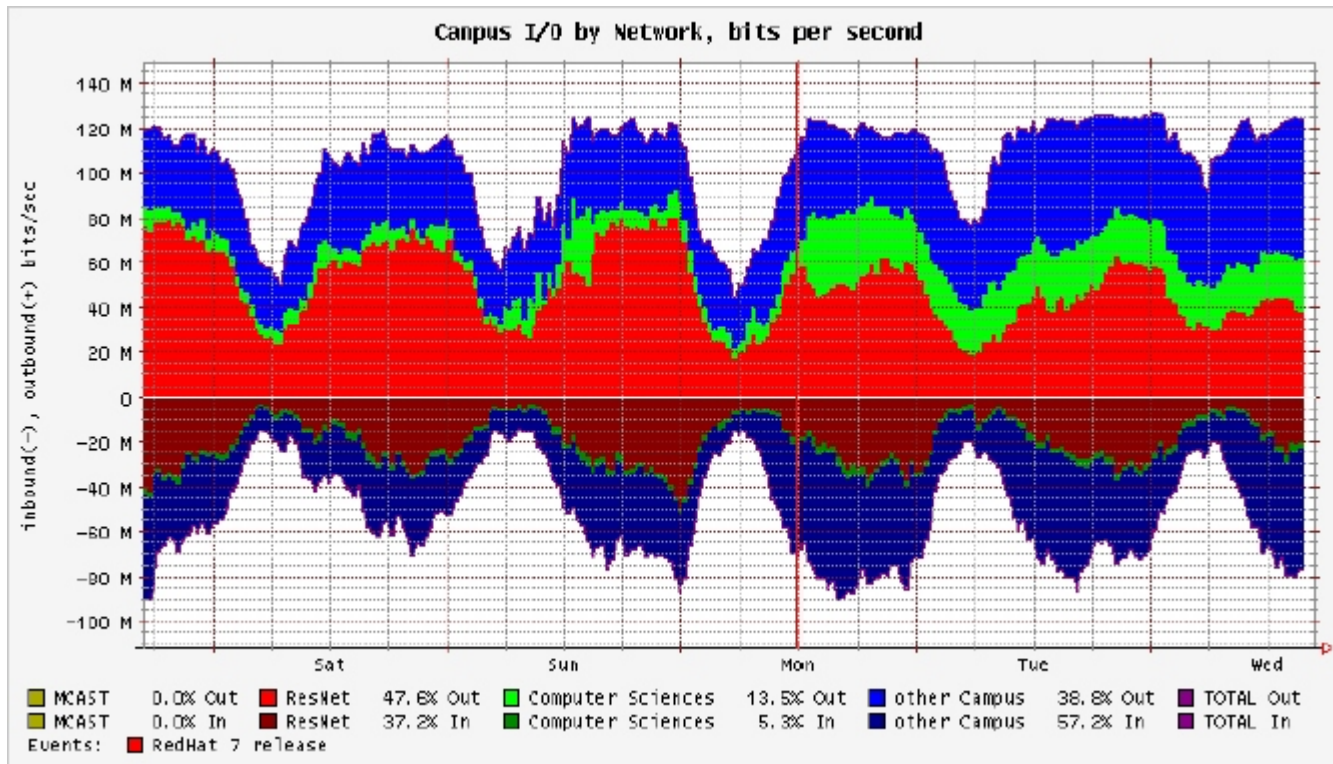
Source: [Barford2001]

Anomaly Identification



‰ Flash crowd anomalies

‰ Caused by software releases or special interest in a web site



Source: [Barford2001]

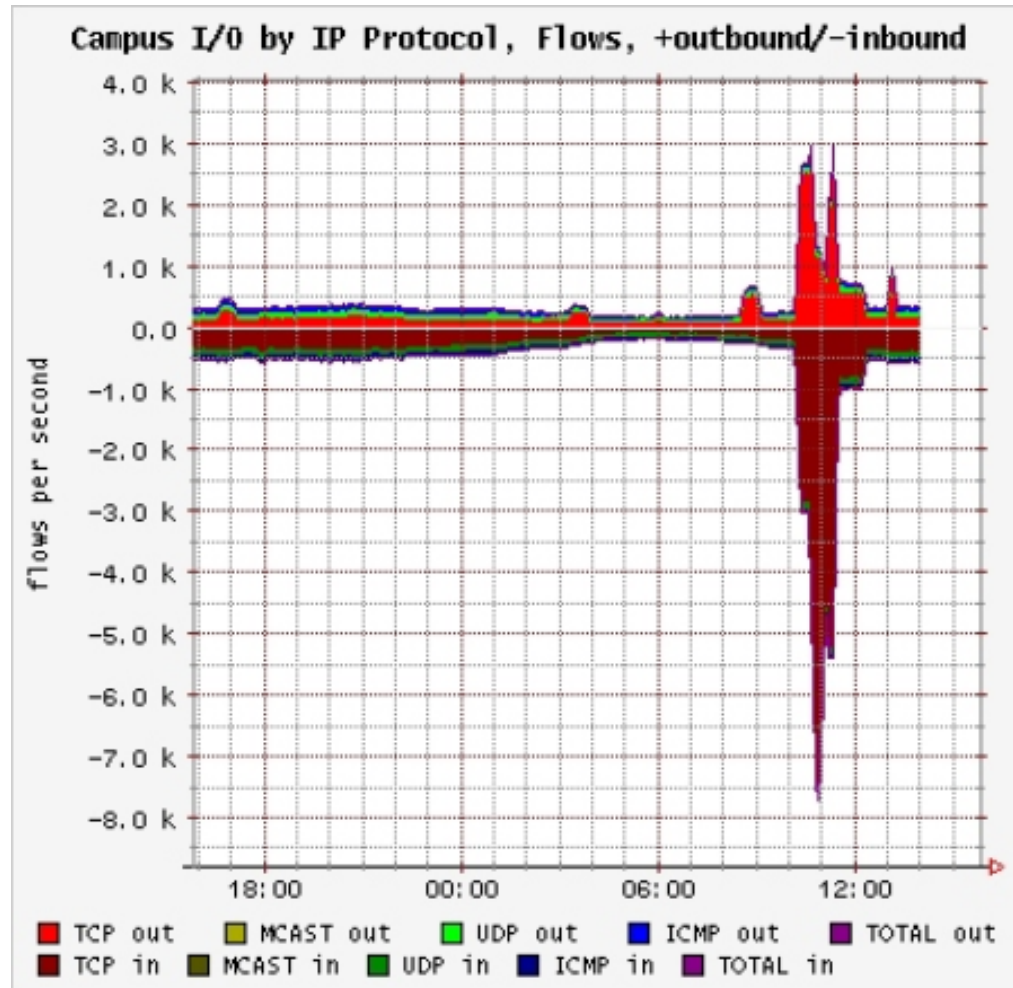
Anomaly Identification



‰ Network abuse anomalies

‰ DoS flood attacks

‰ Port scans

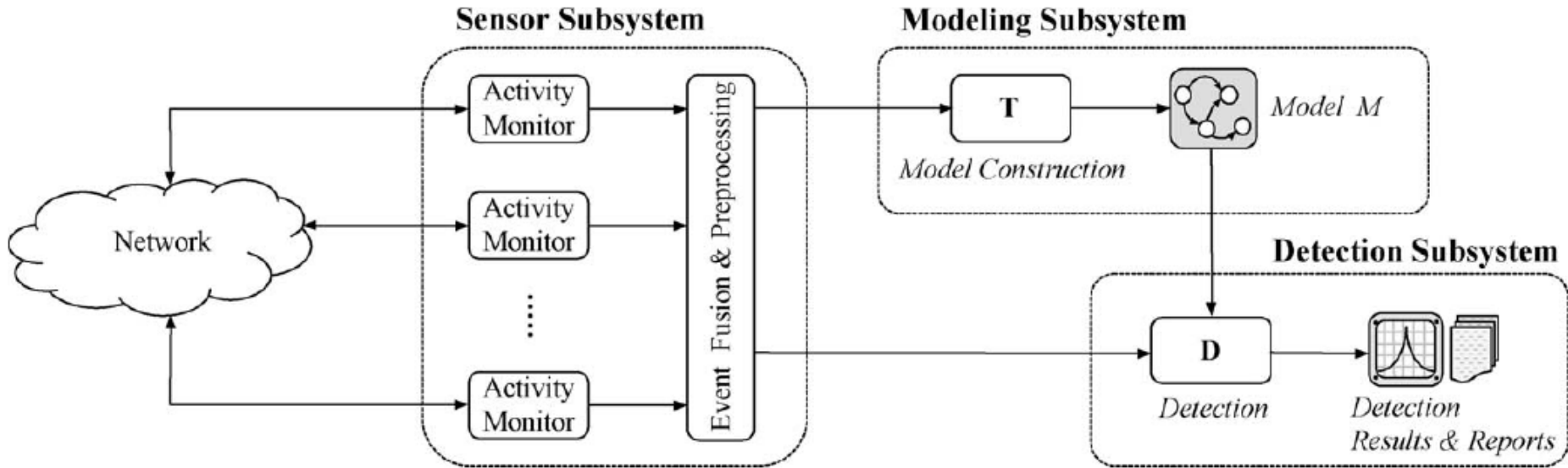


Source: [Barford2001]

Anomaly Detection System

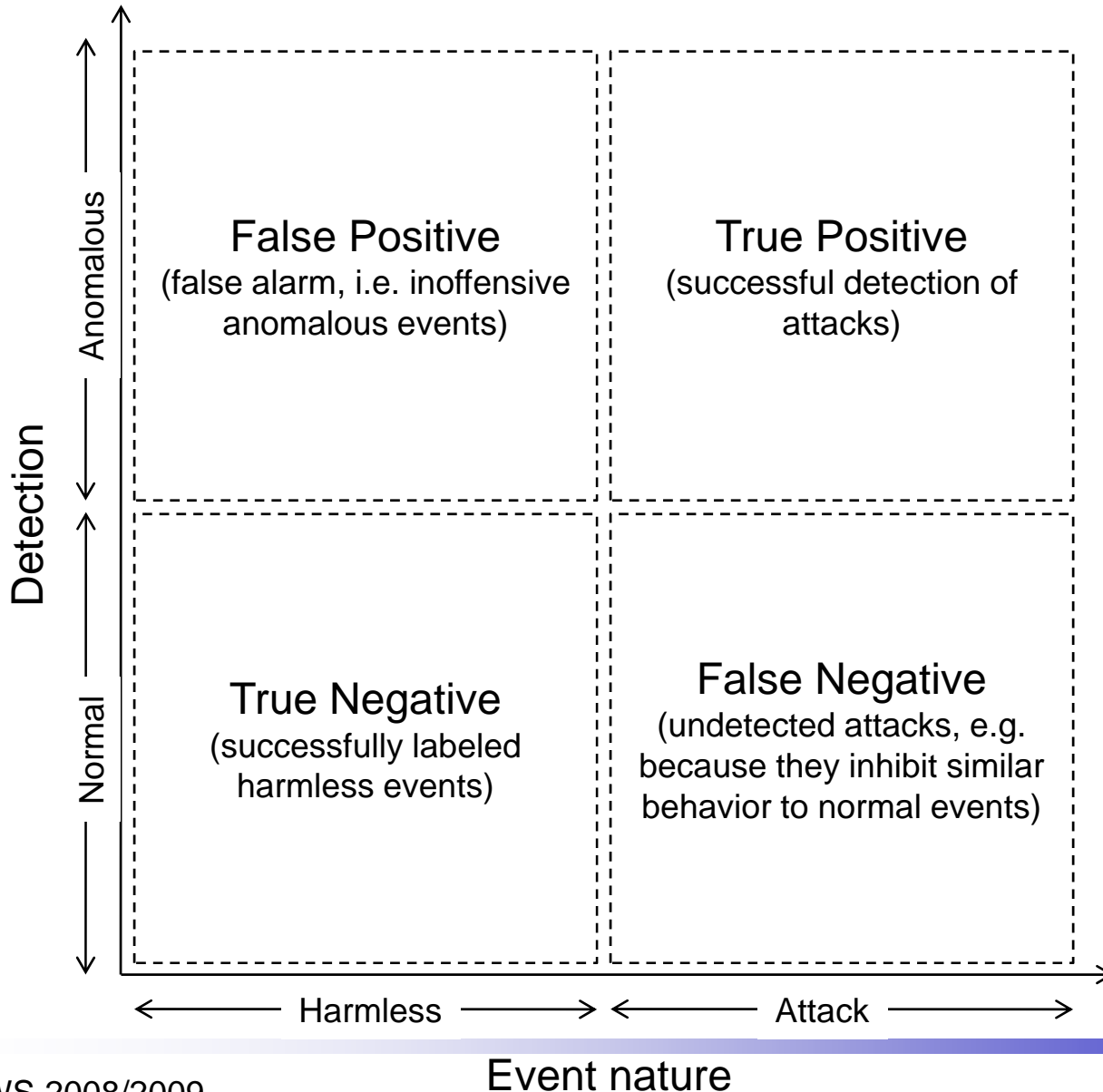


‰ Generic anomaly detection system

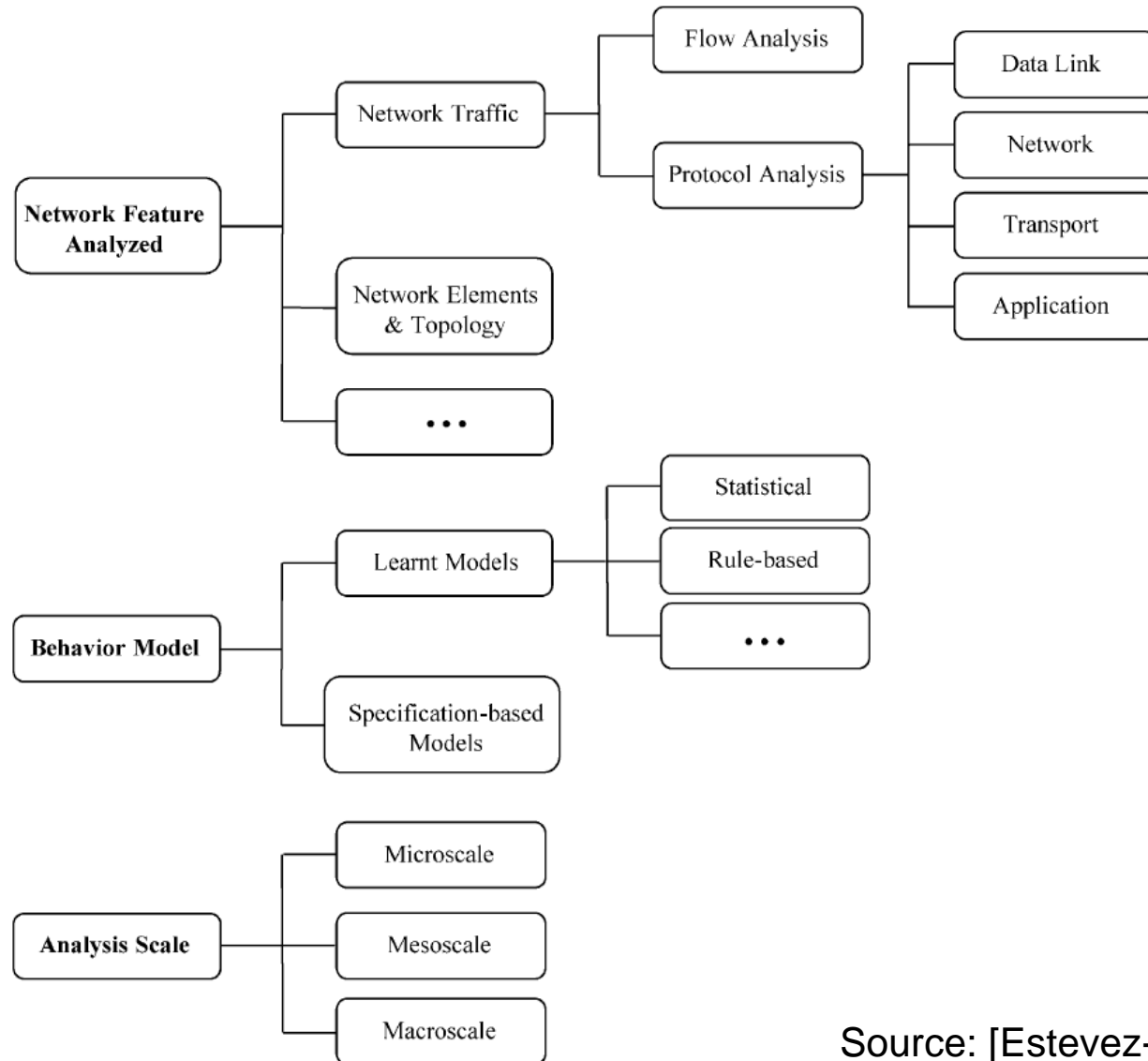


Source: [Estevez-Tapiador2004]

Detection Quality



Classification Criteria



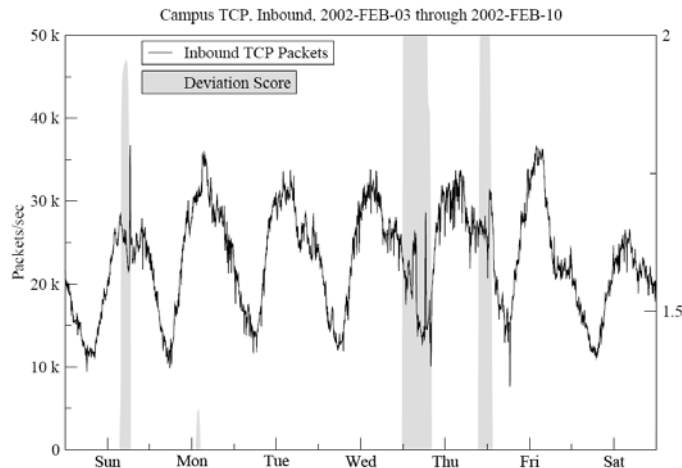
Source: [Estevez-Tapiador2004]

Anomaly Detection Methodologies



‰ Signal analysis

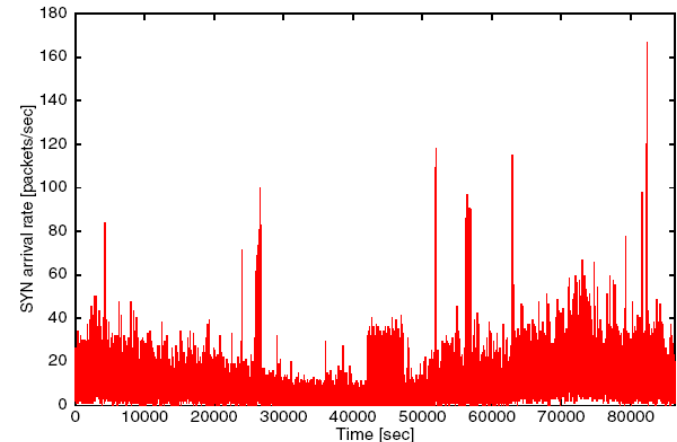
- ‰ Based on the calculation of a deviation score



Source: [Barford2002]

‰ Statistical traffic modeling

- ‰ Analysis of network statistics, e.g. TCP-SYN rate



Source: [Ohsita2004]

‰ Immunological approach

- ‰ Goal: scalable distributed intrusion detection
- ‰ Negative detection with censoring
- ‰ Partial matching with permutation masks
- ‰ Detectors are generated randomly and censored (deleted) if they match normal patterns

PHAD



‰ Packet Header Anomaly Detection (PHAD) [Mahoney2001]

‰ Protocol analysis

“learns” normal ranges of values for each header field
(link, network, transport layer)

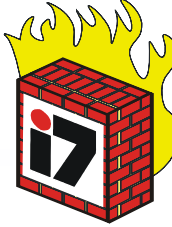
$$\text{score}_{\text{field}} = t * n / r$$

t ... time since previous anomaly

n ... number of observations

r ... number of distinct values

‰ Learning phase + detection phase



- ‰ Application Layer Anomaly Detection (ALAD) [Mahoney2002]

- ‰ Extension to PHAD

- ‰ Five models:

- ‰ $P(\text{src IP} \mid \text{dest IP})$

- Learns normal set of clients for each host, i.e. the set of clients allowed on a restricted service

- ‰ $P(\text{src IP} \mid \text{dest IP}, \text{dest port})$

- Like (1), but one model for each server on each host

- ‰ $P(\text{dest IP}, \text{dest port})$

- Learns the set of local servers which normally receive requests

- ‰ $P(\text{TCP flags} \mid \text{dest port})$

- Learns the set of TCP flags for all packets of a particular connection

- ‰ $P(\text{keyword} \mid \text{dest port})$

- Examines the text in the incoming request (first 1000 bytes)



- ‰ Intrusion Detection Message Exchange Format (IDMEF)

- ‰ RFC 4765

- ‰ Object-oriented approach

- ‰ XML-based encoding

“The purpose of the Intrusion Detection Message Exchange Format (IDMEF) is to define data formats and exchange procedures for sharing information of interest to intrusion detection and response systems, and to the management systems which may need to interact with them.”



%o Message types

- %o Heartbeat message

- %o Alert message

%o Event report

- %o Analyzer – entity which emitted the alert

- %o Classification – what attack has been detected

- %o Source – any combination of multiple objects describing a network node, an user, a process, or a service

- %o Target – any combination of multiple objects describing a network node, an user, a process, a service, or a file

- %o Assessment – severity of the attack and confidence of the analyzer about the validity of the alert

- %o Additional information in (name, value) pairs

IDMEF Example



```
alert TCP any any -> any 80 (msg: "IDS297/http-directory-traversal1"; flags:
AP; content: "../"; reference: arachNIDS,IDS297; idmef: default;)
```

```
<IDMEF-Message version="0.1">
  <Alert alertid="329440" impact="unknown" version="1">
    <Time>
      <ntpstamp>0x3a2d8b3a.0x0</ntpstamp>
      <date>2000-12-05</date>
      <time>16:41:30</time>
    </Time>
    <Analyzer ident="IDS1">
      <Node category="dns">
        <location>San_Francisco_Network</location>
        <name>supersnort</name>
        <Address category="ipv4-addr">
          <address>123.234.123.12</address>
        </Address>
      </Node>
    </Analyzer>
```

IDMEF Example



```
alert TCP any any -> any 80 (msg: "IDS297/http-directory-traversal11"; flags:
AP; content: "../"; reference: arachNIDS,IDS297; idmef: default;)
```

```
<Classification origin="vendor-specific">
  <name>IDS297/http-directory-traversal11</name>
  <url>http://www.whitehats.com/info/IDS297</url>
</Classification>
<Source spoofed="unknown">
  <Node>
    <Address category="ipv4-addr">
      <address>222.222.111.11</address>
    </Address>
  </Node>
</Source>
```

IDMEF Example



```
alert TCP any any -> any 80 (msg: "IDS297/http-directory-traversal1"; flags:
AP; content: "../"; reference: arachNIDS,IDS297; idmef: default;)
```

```
<Target decoy="unknown">
  <Node category="dns">
    <location>San_Francisco_Network</location>
    <Address category="ipv4-addr">
      <address>123.234.123.7</address>
    </Address>
  </Node>
  <Service ident="0">
    <dport>80</dport>
    <sport>1397</sport>
  </Service>
</Target>
<AdditionalData meaning="Packet Payload" type="string">GET
../../stuff/I/shouldnt/be/seeing</AdditionalData>
</Alert>
</IDMEF-Message>
```



IDS – Selected (Open Source) Examples



%o Rule Based Intrusion Detection – Open Source IDS

- %o Support for Windows, UNIX, Linux,...
- %o Huge number of predefined rules
- %o Daily community rules update
- %o Rule set can be edited individually



- %o Three step processing of captured information (capturing via libpcap):
 - %o Preprocessing (normalized and reassembled packets)
 - „ Supports packet de-fragmentation, protocol decoding, state inspection
 - %o Detection Engine works on the data and decides what action should be taken
 - %o Action is taken (log, alert, pass)
 - „ Possible reactions: TCP reset, ICMP unreachable, configuration of firewalls, alerting via email, pager, SMS (plugins)
 - „ Reporting into: Logfiles, LogServer, Database



‰ Snort as Intrusion Prevention System (IPS)

‰ IPTables inserts packets into a queue

‰ Snort receives packets from queue. In case of overload packets are dropped

‰ Preprocessing of data (normalization, reassembly, ...)

‰ Scan engine performs pattern detection upon the data delivered by the preprocessor

„ Possible algorithms

– Wu Manber – Fastest algorithm

– Boyer More – Good for small rulesets

– Aho-Corasick – Good performance even in worst case

‰ After detecting an intrusion the corresponding action is taken

‰ Snort-Inline has is capable to make the packet filter to drop packets, close connections...

‰ Also reconfigures (commercial) firewalls



‰ Bro IDS [Paxson1999]

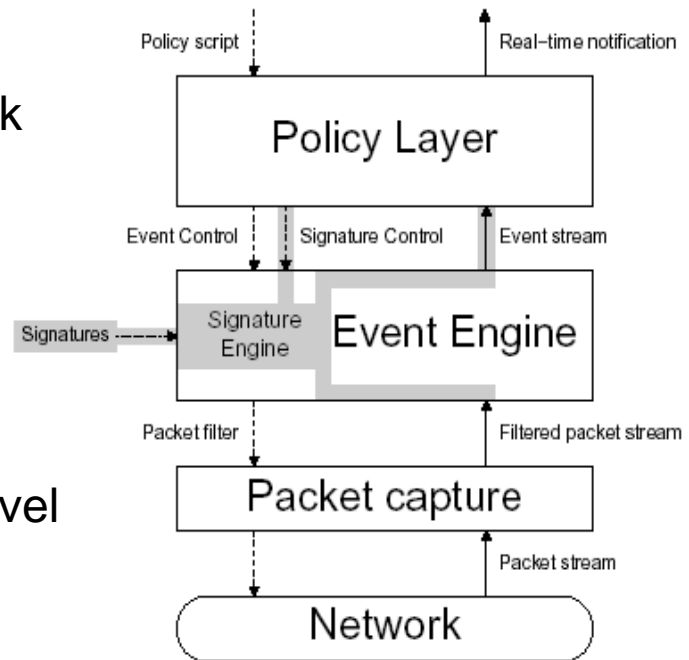
- ‰ Available for Unix and Linux
- ‰ Signature based Intrusion Detection (can work with SNORT Rules)
- ‰ Signatures can be edited individually
- ‰ High number of predefined signatures
- ‰ Reporting: log files, log hosts, via email
- ‰ Saves captured data into libpcap compatible files
- ‰ Supports packet defragmentation, protocol decoding, state inspection
- ‰ Reaction possibilities: connection reset, reconfiguration of firewalls
- ‰ No graphical administration or analysis tools available

Bro



‰ Detailed view:

- ‰ Bro uses several steps to process data
- ‰ Amount of data is reduced in every step
- ‰ The less data has to be processed the more complex actions can be taken
- ‰ Libpcap is used to capture data from network
- ‰ Packet filter removes all packets that are not examined
- ‰ Event engine does some first examinations
 - „ event is created if: header check failed,...
 - „ packet defragmentation is done on this level
- ‰ Signature engine is used to define reoccurring events
- ‰ Policy engine processes all events created by event engine



[Paxson1999]

Bro



- %o Event layer only knows that something has happened but not what-
- %o Bro signatures make use of regular expressions for being able to detect variations of a certain intrusion
 - %o example of a Bro signature to detect variations of the formmail shell command exploit:

```
signature formmail-cve-1999-0172 {  
  „ ip-proto == tcp  
  „ dst-ip == 1.2.0.0/16  
  „ dst-port = 80  
  „ http /. *formmail.*\?. *recipient=[^&]*[;|]/  
  „ event "formmail shell command"  
  „ }
```
- %o Bro uses a **scripting language**, especially designed to facilitate network-traffic analysis and to detect anomalies, which is highly flexible (implicit typing,...)



Prelude Hybrid IDS [PreludeIDS]:

- ‰ Open-source intrusion detection and response system
- ‰ Characteristics:
 - ‰ hybrid: supports network-based (NIDS) and host-based detection (HIDS)
 - ‰ signatures for knowledge-based detection
 - ‰ distributed, hierarchical architecture of sensors and managers
 - ‰ available for many UNIX derivatives
 - ‰ communication between different components based on IDMEF
 - ‰ IDMEF alerts stored in central SQL database
 - ‰ Integration of other open-source packages like Snort, Bro, Honeyd etc.

Prelude IDS



‰ Sensors:

- ‰ NIDS: analysis of captured packets at different protocol layers, port scan detection, ARP spoofing detection, and others
- ‰ HIDS: LML (Log Monitoring Lackey) - analysis of log files on routers, firewalls, and end-systems, file integrity check

‰ Managers:

- ‰ receive IDMEF messages from sensors
- ‰ alert processing: aggregation, correlation, output, storage in database, triggering of response actions (under development)

D-WARD

DDoS Network Attack Recognition and Defense [Mirkovic2003]:

- %o Idea: detect and rate-limit misbehaving flows at the source, i.e. at the ingress point to the network (source-end detection, “firewall for outgoing flows”)

- %o Two levels of detection:

- %o flows = aggregate of packets directed to common destination
- %o individual UDP/TCP connections

- %o Anomaly detection based on models of well-behaving flows and connections

- %o Response:

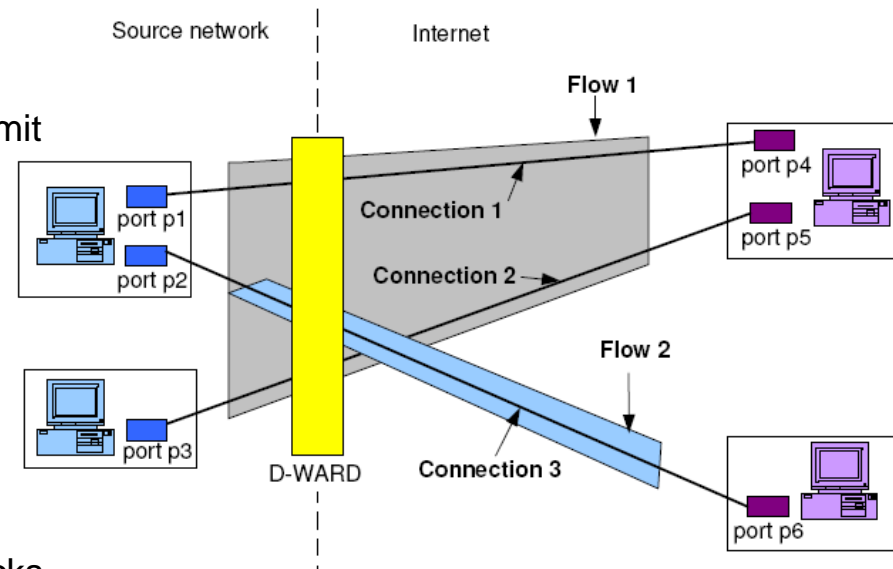
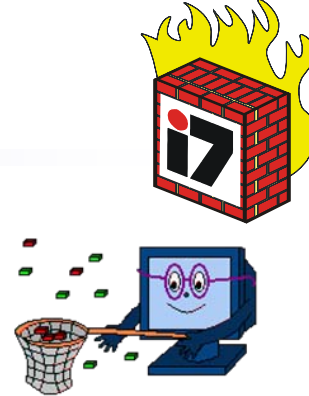
- %o exponential throttling of misbehaving flows
- %o exclude well-behaving connections from ratelimit
- %o gradually remove ratelimit if flow becomes compliant to the model again

- %o Architecture:

- %o Distributed autonomously working systems

- %o Limitations:

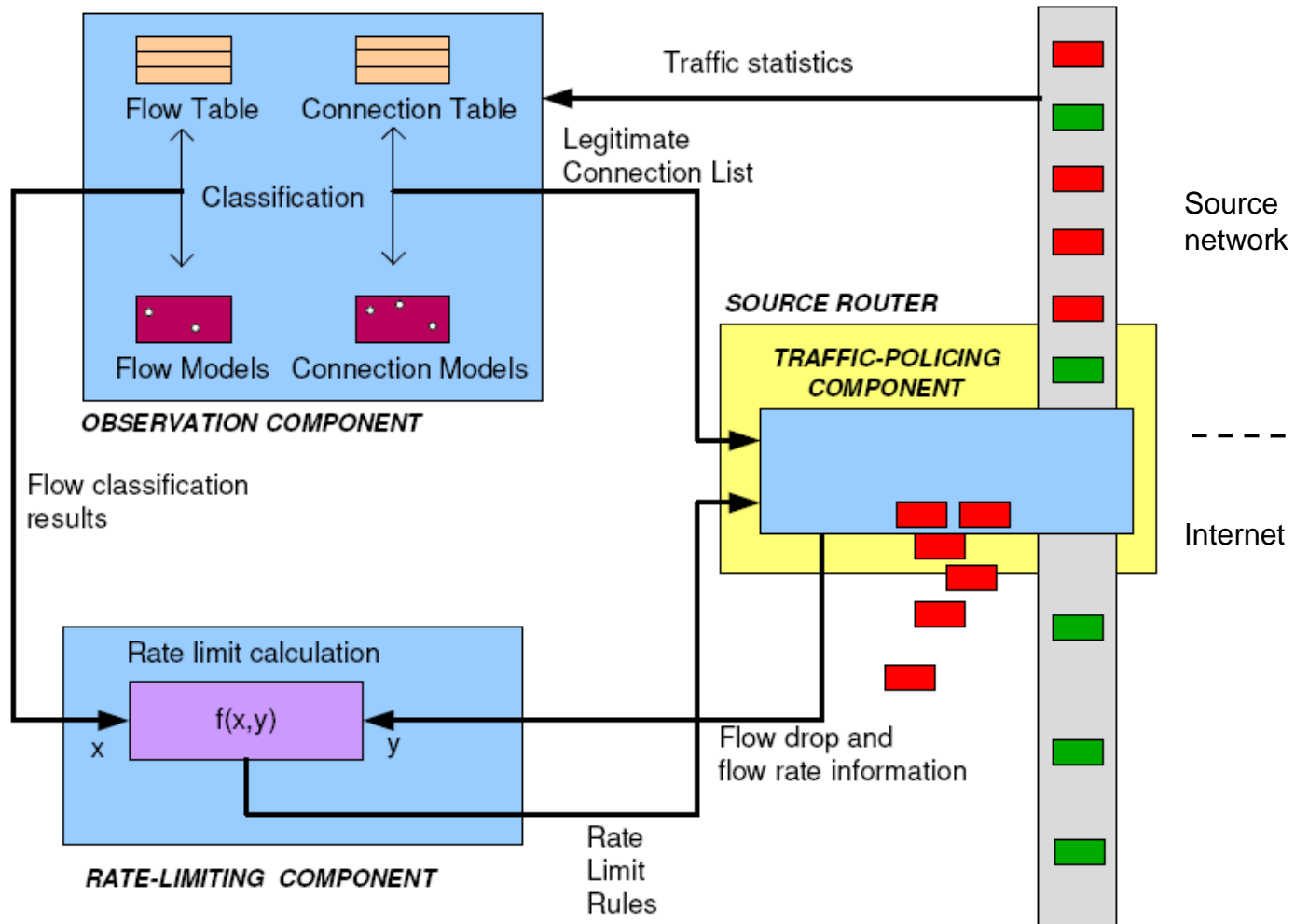
- %o does not work with multi-homed networks and asymmetric routing
- %o anomaly detection based on static, predefined models | no dynamic update
- %o conceived to detect and mitigate flooding attacks
- %o requires large-scale deployment in order to be efficient



D-WARD



% [Mirkovic2003]



DIADEM Firewall



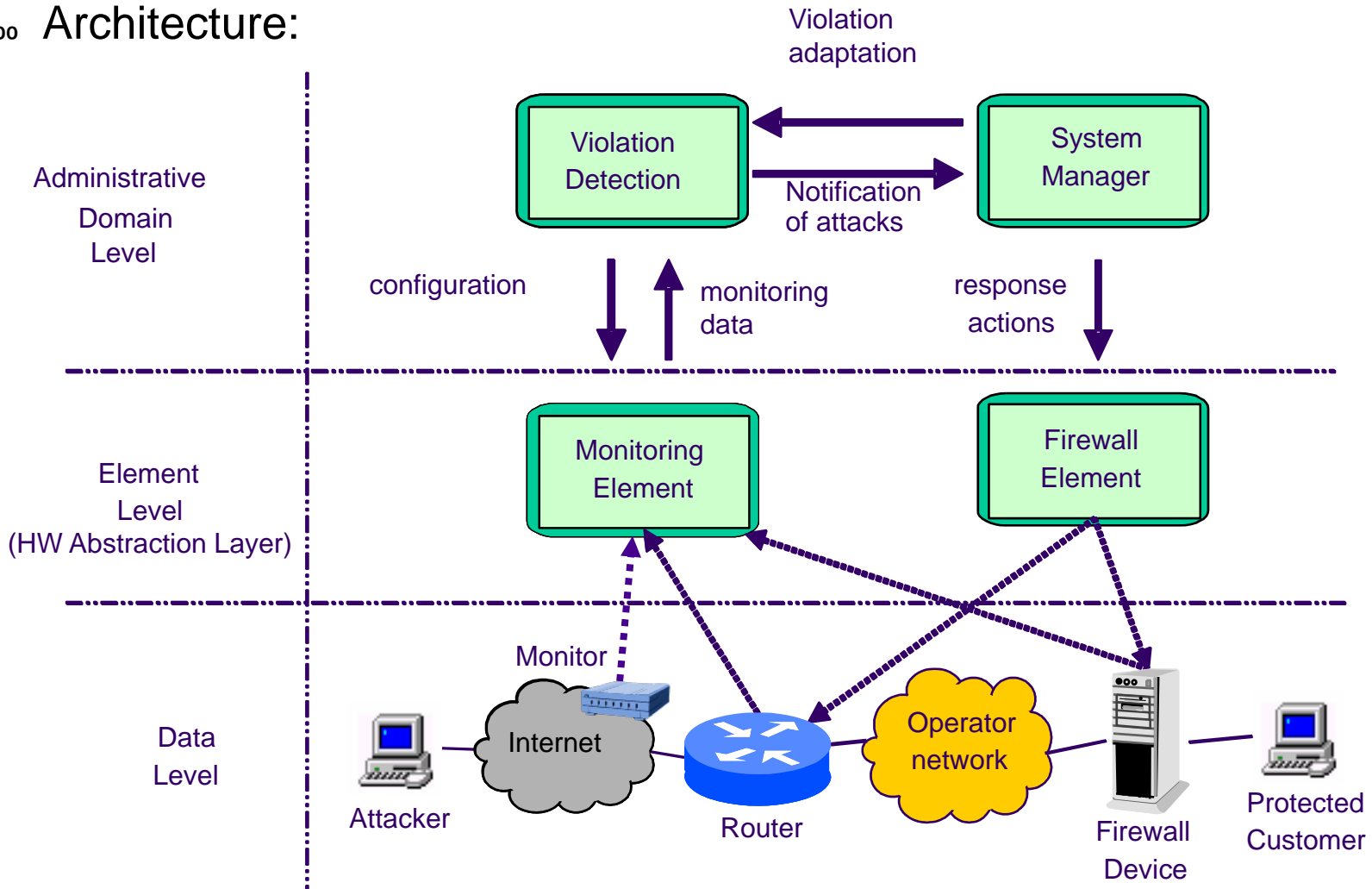
DIADEM Firewall (IST-2002-2.3.1.3) [www.diadem-firewall-org]:

- ‰ Cooperating Autonomous Detection Systems (CATS)
- ‰ Network monitoring environment:
 - ‰ based on IPFIX/PSAMP/Netflow
 - ‰ dynamically reconfigurable according to the current needs for violation detection
- ‰ Violation detection:
 - ‰ distributed and modular
 - ‰ deployment of anomaly and knowledge-based detection methods
- ‰ Firewall elements:
 - ‰ integration of open and commercial high-speed firewalls
 - ‰ possible responses: blocking, rate-limiting, redirection
- ‰ System manager:
 - ‰ defines domain-wide detection and response policies
 - ‰ triggers attack response
- ‰ IDMEF used for event notification between different components
- ‰ Focus on DoS attacks detection and mitigation (flooding attacks, web server overload etc.)

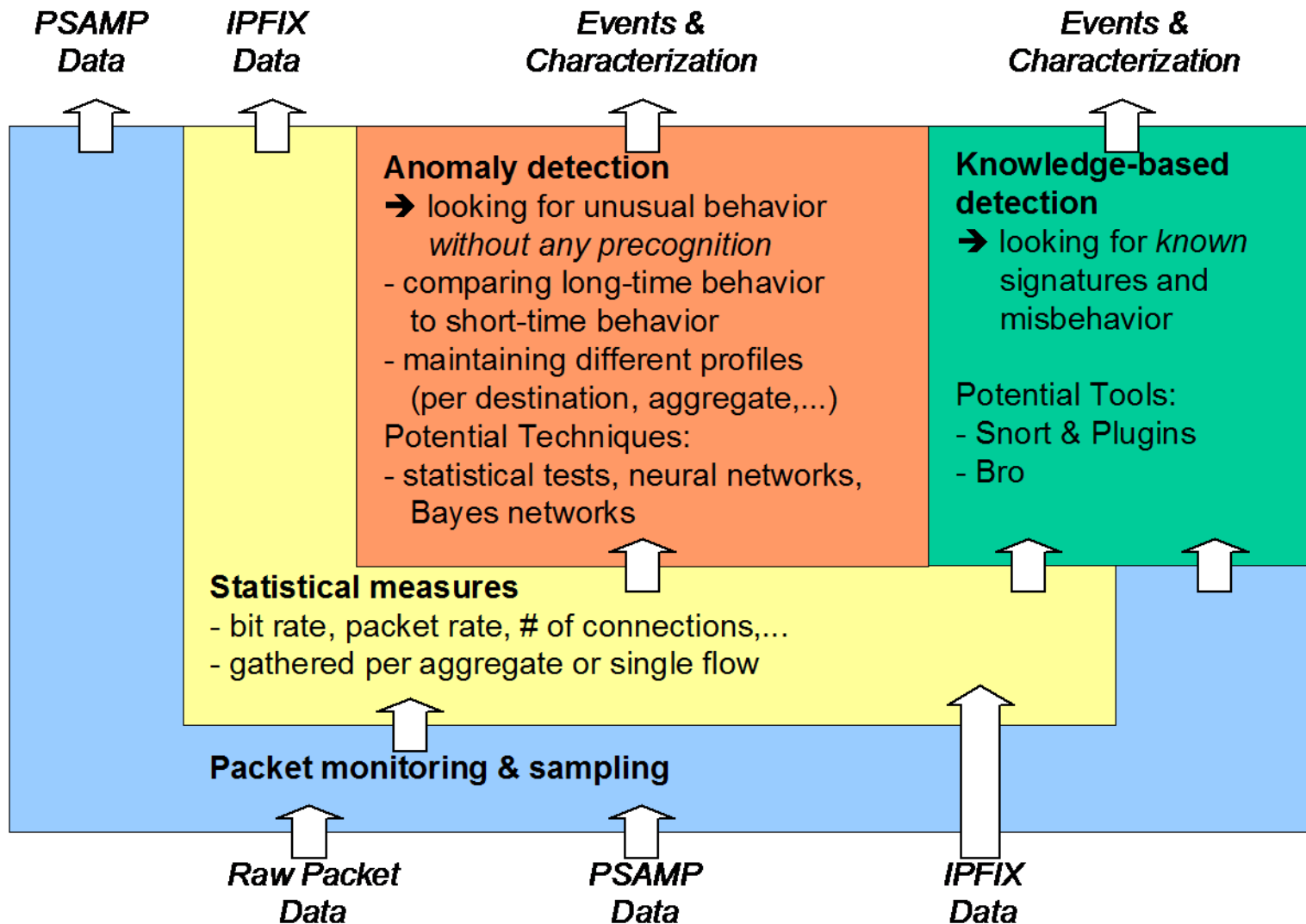
DIADEM Firewall



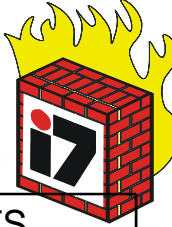
Architecture:



DIADEM Firewall: CATS



Assessment of Distributed Intrusion Detection Systems



		EMER-ALD	Prelude IDS	D-WARD	COSS-ACK	CATS
Attack detection	Local context	yes	yes	yes	yes	yes
	Global context	no (host-based)	no	no	yes	yes
	Knowledge-based detection	yes	yes	no	no	yes
	Anomaly detection	yes	no	yes	yes	yes
Autonomous behavior		no	no	yes	yes	yes
Distributed intelligence	Sep. of monitoring & detection	no	no	no	no	yes
	Distributed detection	yes	partly	no	no	yes

Benchmarking



- ‰ How to measure quality of an IDS?
- ‰ Run a set of attacks within a monitored network segment
 - ‰ How many attacks are detected?
- ‰ Use an increasing level of background traffic in addition to the attacks
 - ‰ 25%, 50%, 75%, 99% of the available bandwidth “filled” with background traffic
- ‰ Stress the IDS to examine if packets are dropped
 - ‰ large number of small / middle size / large packets
- ‰ Use a high rate of http traffic Æ IDS torture
 - ‰ big number of potential HTTP exploits
 - ‰ big number concurrent of http-sessions force the IDS to track these sessions
 - ‰ high rate of new connections
 - ‰ high transaction delay
 - ‰ packet fragmentation

Testing and Benchmarking



- ‰ DARPA Environment (1998/1999)
 - ‰ First systematic effort to test an IDS
 - ‰ Analysis of huge amounts of data, e.g. from Hanscom Air Force Base

- ‰ LARIAT Environment (2000)
 - ‰ Lincoln Adaptive Real-time Information Assurance Test-bed
 - ‰ Emulates network traffic from a small organization
 - ‰ Traffic generation using defined service models

- ‰ Predominant open source philosophy for testing an IDS
 - ‰ Individual test environment
 - ‰ Search for existing exploits / attacks
 - ‰ Mix of background traffic and attack traffic
 - ‰ Analysis of the detection ratio (false positive / false negative)

Source: [Athanasiaades2003]

Summary (what do I need to know)



- ‰ Principles of Intrusion Detection Systems

- ‰ Categories

- ‰ Knowledge-based / signature-based

- ‰ Anomaly detection

- ‰ Host-based IDS vs. network IDS

- ‰ Detection quality

- ‰ False positives

- ‰ False negatives

Additional References



- [Barford2001] P. Barford and D. Plonka, "Characteristics of Network Traffic Flow Anomalies," Proceedings of ACM SIGCOMM Internet Measurement Workshop, October 2001.
- [Barford2002] P. Barford, J. Kline, D. Plonka, and A. Ron, "A Signal Analysis of Network Traffic Anomalies," Proceedings of ACM SIGCOMM Internet Measurement Workshop, Marseilles, France, November 2002.
- [Cabrera2000] J. B. D. Cabrera, B. Ravichandran, and R. K. Mehra, "Statistical Traffic Modeling for Network Intrusion Detection," Proceedings of 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2000, pp. 466.
- [Caswell2004] B. Caswell and J. Hewlett, "Snort Users Manual," The Snort Project, Manual, May 2004. (http://www.snort.org/docs/snort_manual.pdf)
- [Estevez-Tapiador2004] J. M. Estevez-Tapiador, P. Garcia-Teodoro, and J. E. Diaz-Verdejo, "Anomaly detection methods in wired networks: a survey and taxonomy," Computer Communications, vol. 27, July 2004, pp. 1569-1584.
- [Hofmeyer1998] S. Hofmeyer, S. Forrest, and P. D'haeseleer, "An Immunological Approach to Distributed Network Intrusion Detection," Proceedings of First International Workshop on the Recent Advances in Intrusion Detection (RAID'98), Louvain-la-Neuve, Belgium, September 1998.
- [Hussain2003] A. Hussain, J. Heidemann, and C. Papadopoulos, "A Framework for Classifying Denial of Service Attacks," Proceedings of ACM SIGCOMM Conference, Karlsruhe, Germany, August 2003, pp. 99-110.
- [Kemmerer2002] R. Kemmerer and G. Vigna, "Intrusion Detection: A Brief History and Overview," IEEE Computer - Special Issue on Security and Privacy, April 2002, pp. 27-30.
- [Mahoney2001] M. V. Mahoney and P. K. Chan, "PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic," Florida Tech., Technical Report CS-2001-4, 2001.
- [Mirkovic2004] J. Mirkovic and P. Reiher, "A Taxonomy of DDoS Attack and DDoS Defense Mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, April 2004, pp. 39-53.
- [Moore2001] D. Moore, G. M. Voelker, and S. Savage, "Inferring Internet Denial-of-Service Activity," Proceedings of USENIX Security Symposium, Washington, DC, August 2001.
- [Wang2002] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN Flooding Attacks," Proceedings of IEEE INFOCOM 2002, 2002.