Protection of Communication Infrastructures

Chapter 7
Intrusion Detection Systems

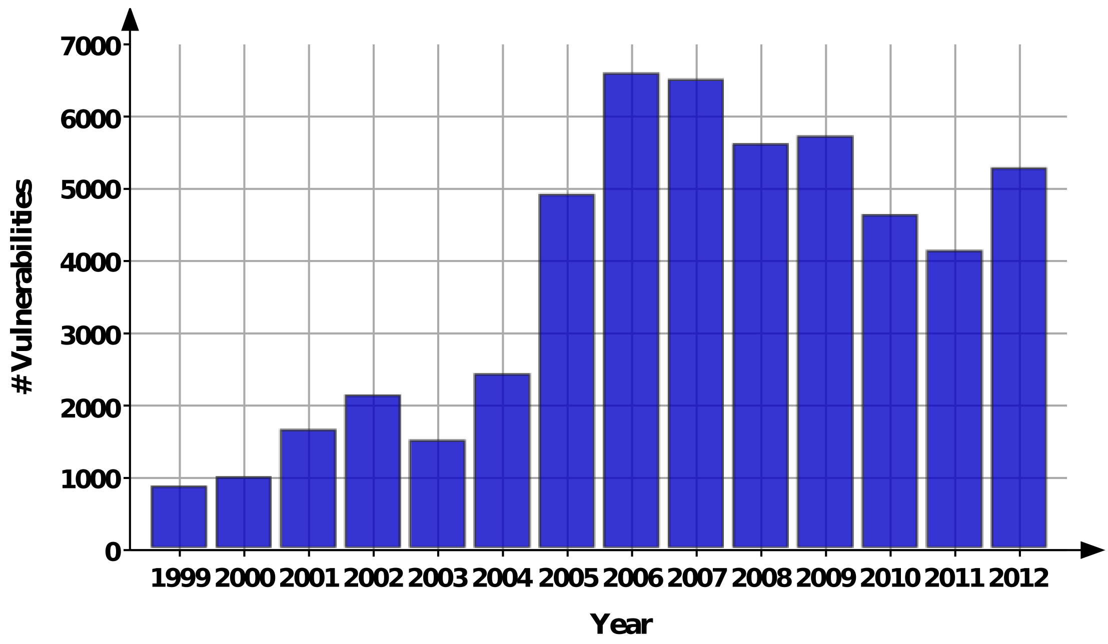
- Motivation
- Goals and Tasks of an IDS
- NIDS types & properties
- Intrusion Prevention
- Evading IDS

(Acknowledgement: some of slides have been adapted from [CDS05, Kön03])

Introduction

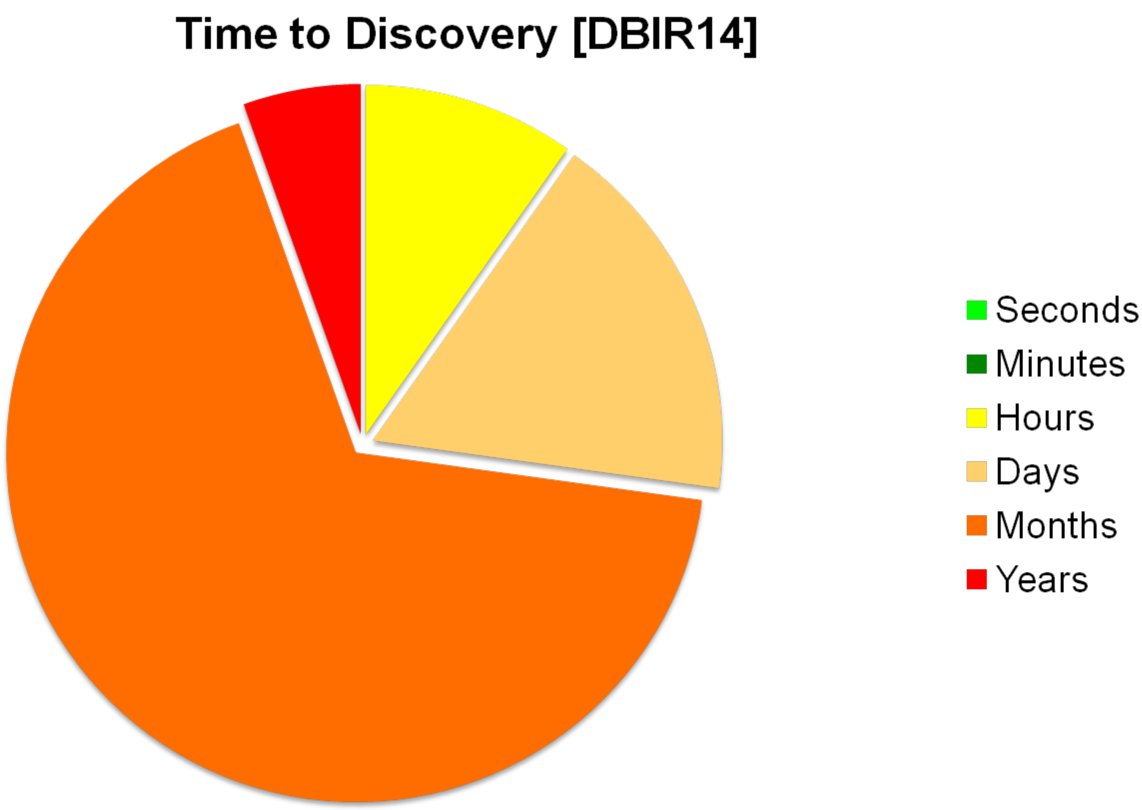
- Definition:
  - An *intrusion* is an action or set of actions aimed at compromising the confidentiality, integrity or availability of a service or system
- Principal defense categories:
  - Prevention
  - Detection
  - Response

Number of vulnerabilities reported per year (CVE)

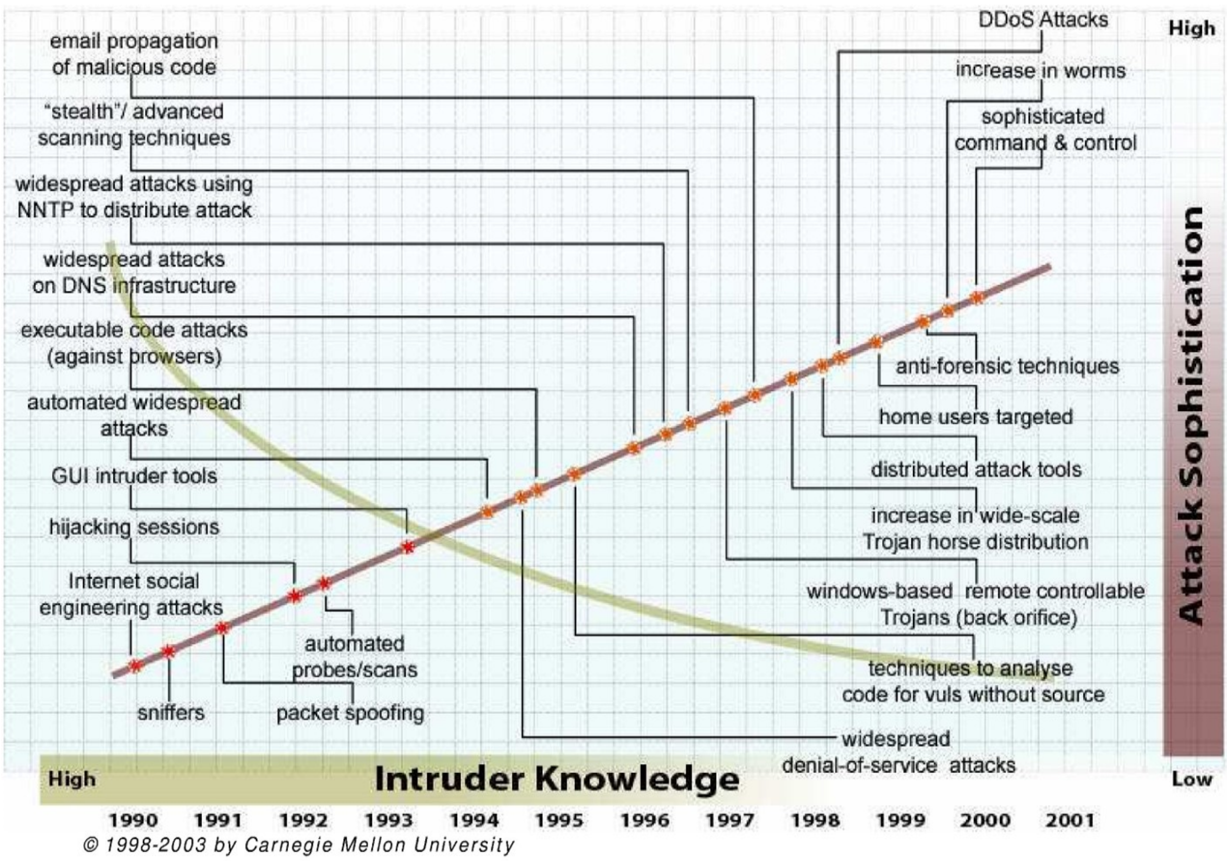


- These numbers are just a trend indicator, as:
  - not all of vulnerabilities are found and published, and
  - not all vulnerabilities receive a CVE number

How long to discover a case of cyber-espionage?



Attack Sophistication vs. Intruder Knowledge



A Long History of Intrusion Detection

1980 – James Anderson: Computer Security Threat Monitoring and Surveillance

1983 – Dorothy Denning (SRI-International): Analysis of audit trails from government mainframe computers

1984 – Dorothy Denning: Intrusion Detection Expert System (IDES)

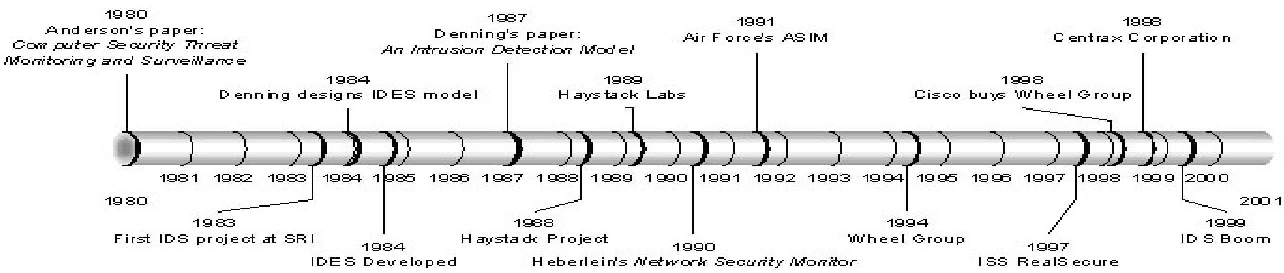
1988 – Lawrence Liverpool Laboratories: Haystack Projekt

1990 – Heberlein: A Network Security Monitor (NSM)

1994 – Wheel Group: First commercial NIDS (NetRanger)

1997 – ISS: Real Secure

Early 2000 - Boom of Intrusion Detection System



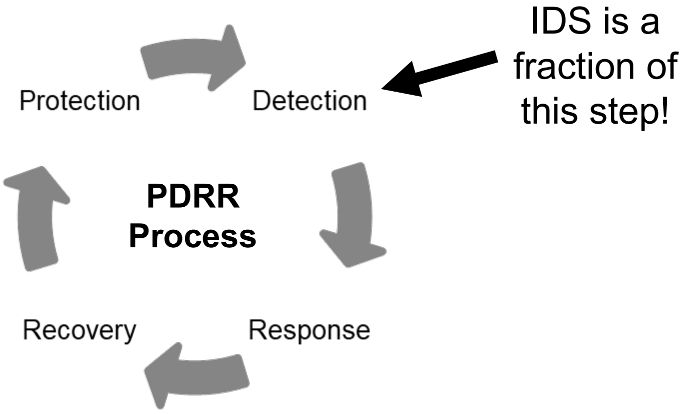
http://www.securityfocus.com/infocus/1514

Goal of Intrusion Detection Systems

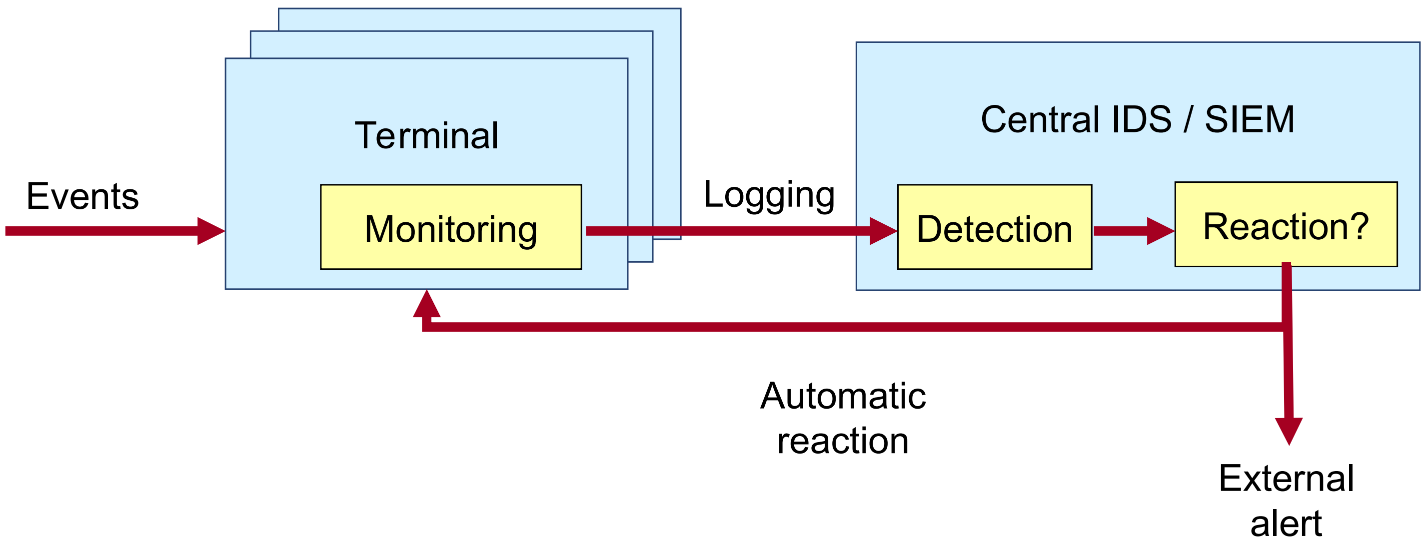
- Overall goal: Supervision of computer systems and communication infrastructures in order to detect intrusions and misuse
- Why detection of attackers?
  - Full protection is usually not possible!
  - Security measures too expensive or with too low flexibility, e.g., not possible to build every functionality in ASICs
  - Wrong postulates about capabilities of attackers (NSA?)
  - Unpatched systems for compliance reasons (medical systems etc.)
  - Because legitimate users get annoyed by too many preventive measures and may even start to circumvent them (introducing new vulnerabilities)
  - Because preventive measures may fail:
    - Incomplete or erroneous specification / implementation / configuration
    - Inadequate deployment by users (just think of passwords...)
- What can be attained with intrusion detection?
  - Detection of attacks and attackers
  - Detection of system misuse (includes misuse by legitimate users)

Possibilities of Intrusion Detection Systems

- Using a detection system only makes sense if there are consequences!
- Possible goals
  - Limitation of damage if (automated) response mechanisms exist
  - Gain of experience in order to recover from attack and improve preventive measures
  - Deterrence of other potential attackers (if and only if police is able to arrest them!)



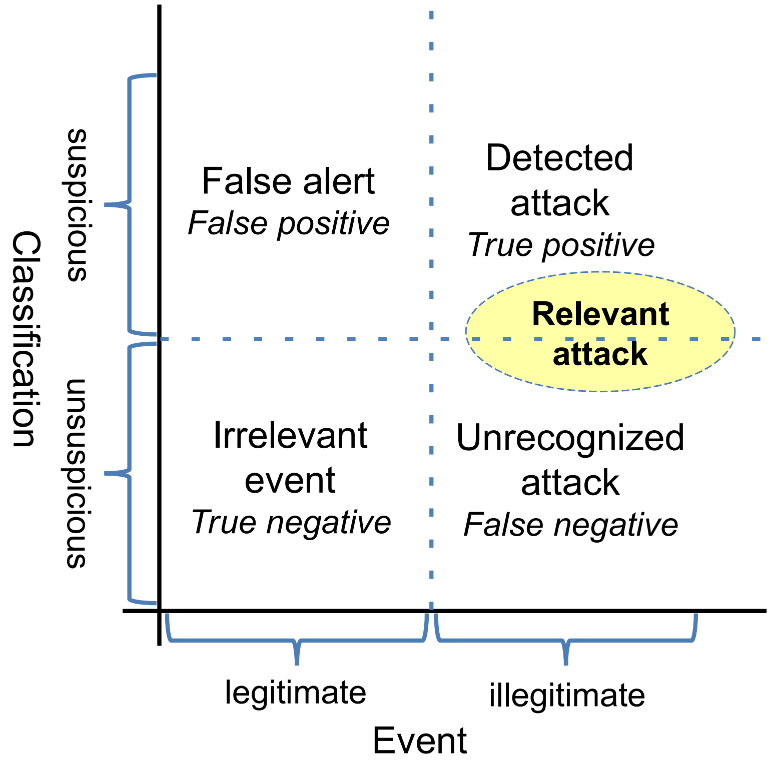
Operation of Intrusion Detection Systems



Tasks of an Intrusion Detection System

- *Audit:*
  - Recording of all security relevant events of a supervised system
  - Preprocessing and management of recorded audit data
- *Detection:*
  - Automatic analysis of audit data
  - Principle Approaches:
    - Signature analysis
    - Abnormal behavior detection (based on knowledge)
    - Anomaly detection (based on learned "normal level")
  - Types of errors:
    - *False positive:* a non-malicious action is reported as an intrusion
    - *False negative:* an intrusion is not detected (a "non-event")
- *Response:*
  - Reporting of detected attacks (alerts)
  - Potentially also initiating countermeasures (reaction)

Detection Quality



Requirements to Intrusion Detection Systems

- High accuracy (= low rate of false positives and false negatives)
- Easy to integrate into a system / network
- Easy to configure & maintain
- Autonomous and fault tolerant operation
- Low resource requirements
- Self protection, so that an IDS itself can not easily be deactivated by a deliberate attack (in order to conceal subsequent attacks)

Classification of Intrusion Detection Systems

- Classification of intrusion detection systems (IDS):
  - Scope:
    - Host-based: analysis of system events
    - Network-based: analysis of exchanged information (IP packets)
    - Hybrid: combined analysis of system events and network traffic
  - Time of analysis:
    - Online analysis
    - Post mortem (Forensic tools, not covered here)

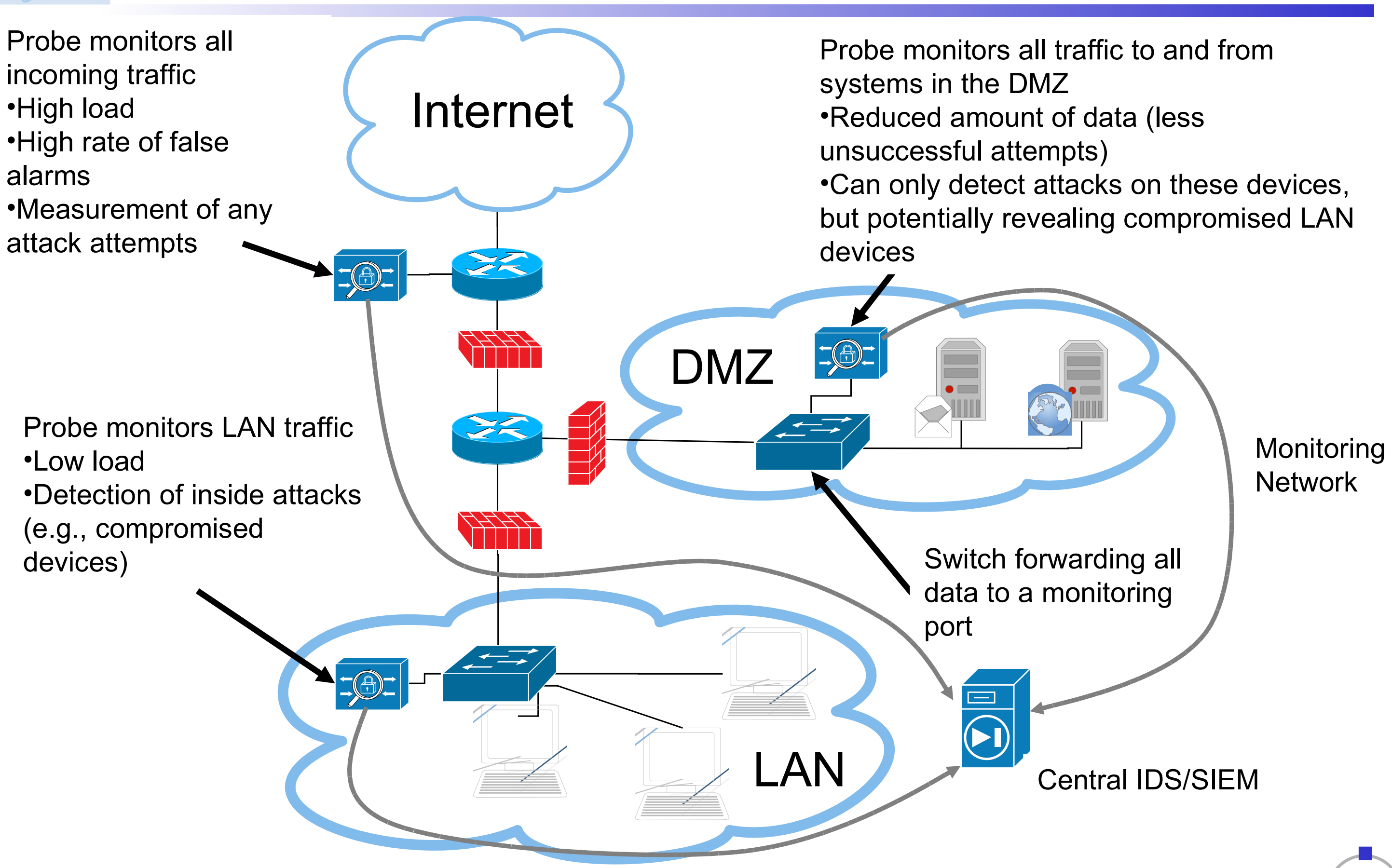
Host Intrusion Detection Systems (HIDS)

- Works on information available on a system:
  - OS and application logs
  - System file modification
  - Illegal file access
  - Login behavior (invalid tries, times)
  - Analysis of system resource consumption
  - Searches for viruses, rootkits etc.
- Can detect attacks by insiders (e.g. copy to USB stick), but:
  - Has to be installed on every system
    - Hard to manage on a large number of systems
    - Not available for every platform (e.g. routers, printers, medical devices etc.)
    - May be disabled by the attacker!
  - Produces lots of (potentially non-useful) information
  - Often no real-time analysis but predefined time intervals

Network Intrusion Detection System (NIDS)

- Analysis of network monitoring information (mostly on network layer)
- Existing systems use a combination of
  - Signature-based detection
  - Deviation from defined protocol behavior (stateful)
  - Statistical anomaly analysis
- Can even detect 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 → Easier to manage & ensure monitoring of all devices
- (Obviously) cannot detect offline attacks, e.g., copy to USB stick
- In reality also produces lots of (potentially non-useful) information
- What about encrypted protocols?
- **We will concentrate on NIDS in the following…**

Placement of a Network Intrusion Detection System



TELEMATIK Rechnernetze

Intrusion Detection Message Exchange Format (1)

- Intrusion Detection Message Exchange Format (IDMEF)
  - IETF Intrusion Detection WG
  - RFC 4765 (Experimental)
  - Defines messages between probes and central components
  - Allows (in principle) to combine devices of different vendors

  - Object-oriented approach
  - XML-based encoding

TELEMATIK Rechnernetze

Intrusion Detection Message Exchange Format (2)

- Message types
  - Heartbeat message
  - Alert message (ToolAlert, OverflowAlert, CorrelationAlert)
  - ...

- Event report
  - Analyzer – entity which emitted the alert
  - Classification – what attack has been detected
  - Source – any combination of multiple objects describing a network node, an user, a process, or a service
  - Target – any combination of multiple objects describing a network node, an user, a process, a service, or a file
  - Assessment – severity of the attack and confidence of the analyzer about the validity of the alert
  - Additional information in (name, value) pairs

Signature-based detection

- Basic idea:
  - Some attack patterns can be described with sufficient detail → specification of “attack signatures”
  - Event generated if packet(s) contains known attack signatures
- Identifying attack signatures:
  - Analyzing vulnerabilities
  - Analyzing past attacks that have been recorded in the audit
- Specifying attack signatures:
  - Based on identified knowledge so-called rules describing attacks are specified
  - Most IDS offer specification techniques for describing rules
  - Achievable detection quality directly dependent on quality of signature database (DB)

Signature-based detection – Example: Snort (1)

- Each detected attack type needs a predefined rule

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any
  (msg: "Ping-of-Death detected";
   dsize: > 10000;
   sid: 3737844653)
```

- Shall detect Ping-of-Death packets, i.e., packets that are unusually large and crash the operating system
- How do these packets look in layer 3 (and below)
  - MTU is usually 1,500 bytes
  - → at least 7 packets!

- Requires preprocessing of packets!

Signature-based detection – Example: Snort (2)

More sophisticated example, checking for mail server buffer overflows:

```
alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25
  (msg:"SERVER-MAIL RCPT TO overflow";
   flow:to_server,established;   <- Quick check
   content:"rcpt to|3A|";        <- Better check (requires TCP reassembly)
   nocase;
   isdataat:256,relative;
   pcre:"/^RCPT TO\x3a\s*\x3c?[^\n\x3e]{256}/im";   <- Very slow reqular expression check
   classtype:attempted-admin;
   sid:654;
   rev:23;)
```


Signature-based detection – Packet Processing

- Three step processing of captured packets:
  - Preprocessing:
    - Normalized and reassembled packets (layer 3)
    - Recovery of TCP data flows (layer 4)
    - Normalization of application layer protocols
  - Detection engine works on the data and decides what action should be taken
  - Action is taken (log, alert, pass)

Signature-based detection – Properties

- Advantages:
  - Easy to setup
  - In some environments acceptable false positive rate
- Drawbacks:
  - Requires prior knowledge of all potential attacks
  - Signature database requires continuous updating
    - Large databases, difficult to maintain
    - Large number of “special plugins” for attacks not to express with rule language, e.g., to detect port scans
  - High false negatives rate if signature DB not adapted or up-to-date
  - IP & TCP preprocessing requires significant resources
  - Possibility of bypassing:
    - Attackers being aware of a certain IDS may try to craft attacks that are not covered by any signature
    - May be tested offline!

Detection of Abnormal Behavior

- Basic idea – detect behavior that differs significantly from normal use:
- Users and systems have “normal” use pattern:
  - Activity pattern
  - Used protocols & protocol states
  - Accessed servers
  - Traffic volumes etc.
- Assumption: “behavior” can be described by an administrator
  - Needs a specification, e.g., in a rule language
  - For generic protocols such a description may be predefined
- Analysis:
  - Events matched against rules
  - Any mavericks will be reported
  - Comparable to a firewall that only performs logging…

Detection of Abnormal Behavior – Example Systems

- NetSTAT [VK98]
  - Early academic example
  - Compares network traffic in probes with fact base
  - Simple application layer inspection, e.g., NFS
- StealthWatch (commercial)
  - Commercial system
  - Analyses flow information in switches (e.g. Cisco NetFlow or sFlow)
  - Can detect network scans, worm spreading, DoS attacks …
- Bro Security Monitor
  - Long-living open source project
  - Performs stateful protocol analysis
  - Reports protocol deviations, e.g., undocumented commands
- (Honey pots & honey networks)
  - Systems not accessed by legitimate users by design
  - All access may be considered illegitimate…

Detection of Abnormal Behavior – Properties

- Advantages:
  - Approach can detect unknown attacks
  - Attacks cannot easily be prepared to be not detected
  - If well set up: acceptable false positive rate
  - Events rather easy to interpret
- Drawbacks:
  - High administrative effort
  - Some attacks (e.g. buffer overflows) are most likely not detected

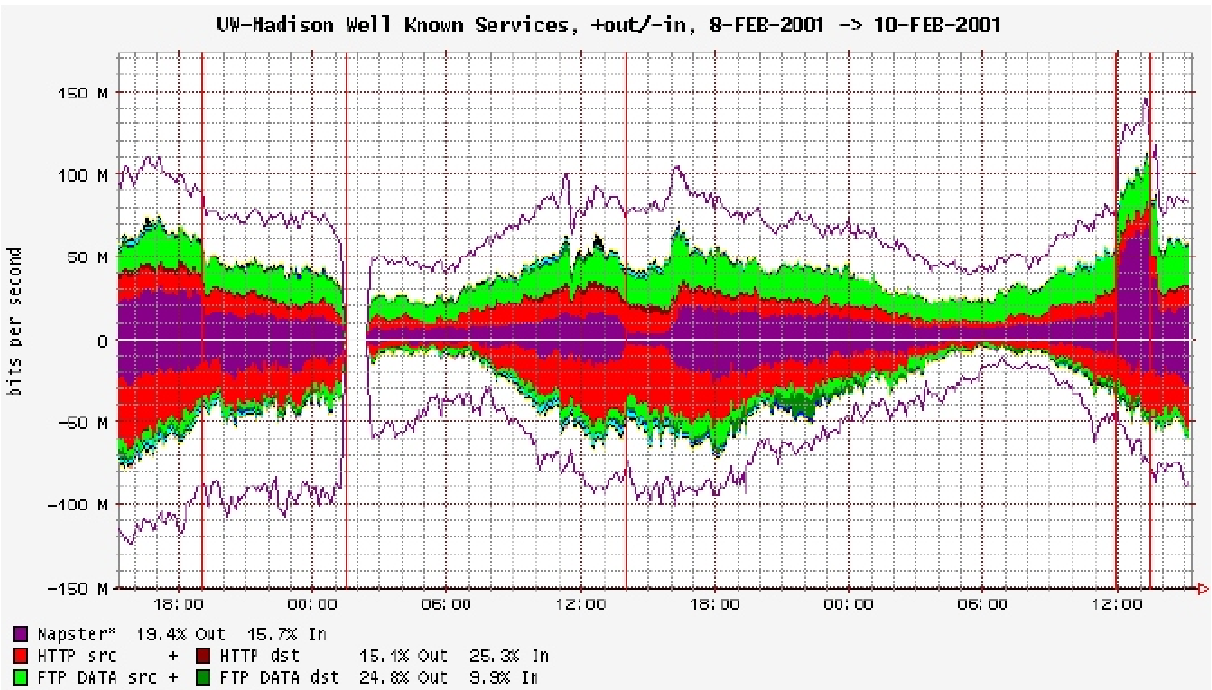
- Direct firewall integration perhaps better…

Automatic Anomaly Detection – Overview

- Basic idea – detect behavior that differs significantly from normal use, which is automatically learned
- Assumption: “normal user behavior” can be described statistically
  - Requires a learning phase / specification of normal behavior
  - Can learn significantly more features than an administrator is able to specify manually!
- Analysis:
  - Compares recorded events with reference profile of normal behavior
  - Use statistics and anomaly detection techniques to find outliers
  - Report if there is a timely correlation of a significant number of outliers

Automatic Anomaly Detection – Example (1)

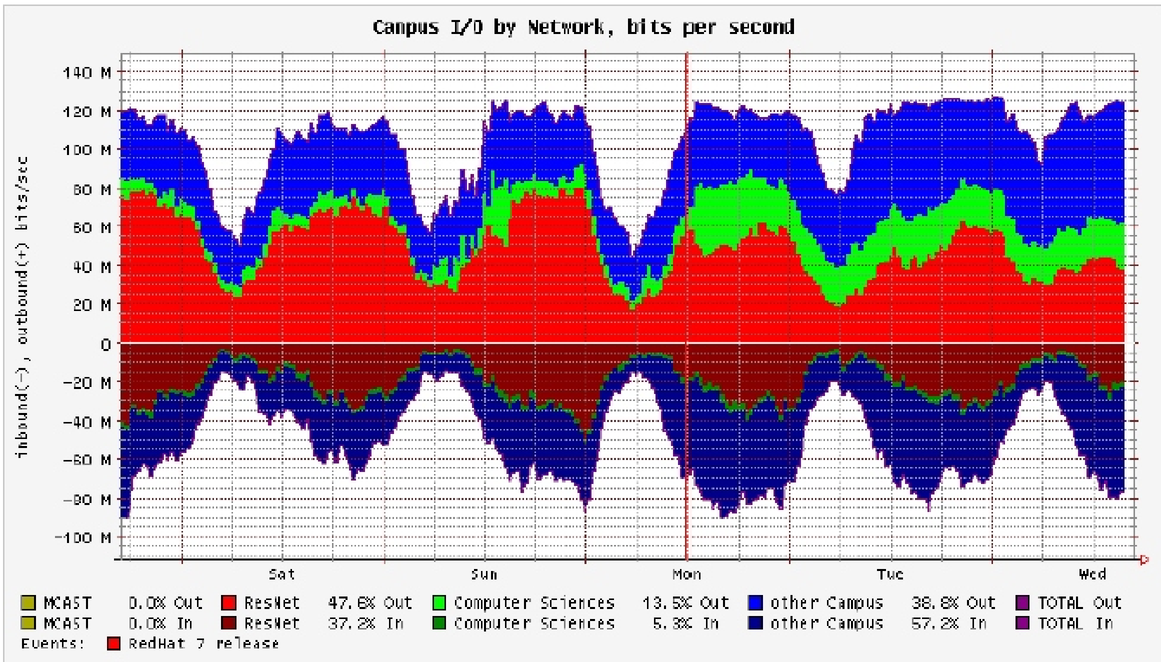
- Network operation anomalies
  - Caused by configuration changes



Source: [Bar01]

Automatic Anomaly Detection – Example (2)

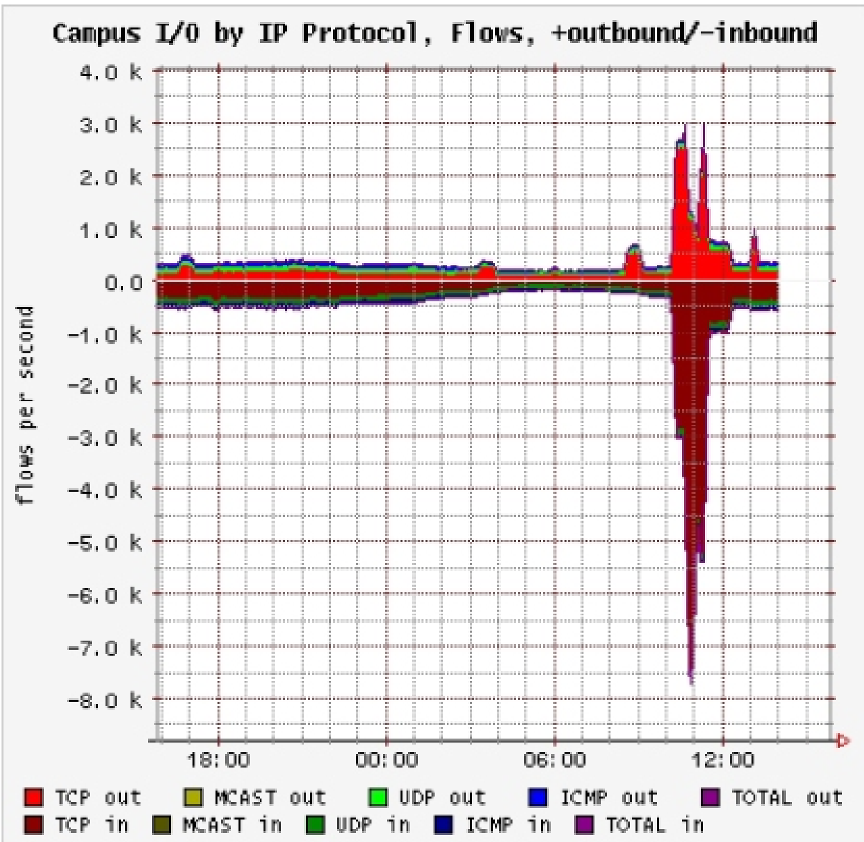
- "Flash crowd anomalies"
  - Caused by software releases or special interest in a web site



Source: [Bar01]

Automatic Anomaly Detection – Example (3)

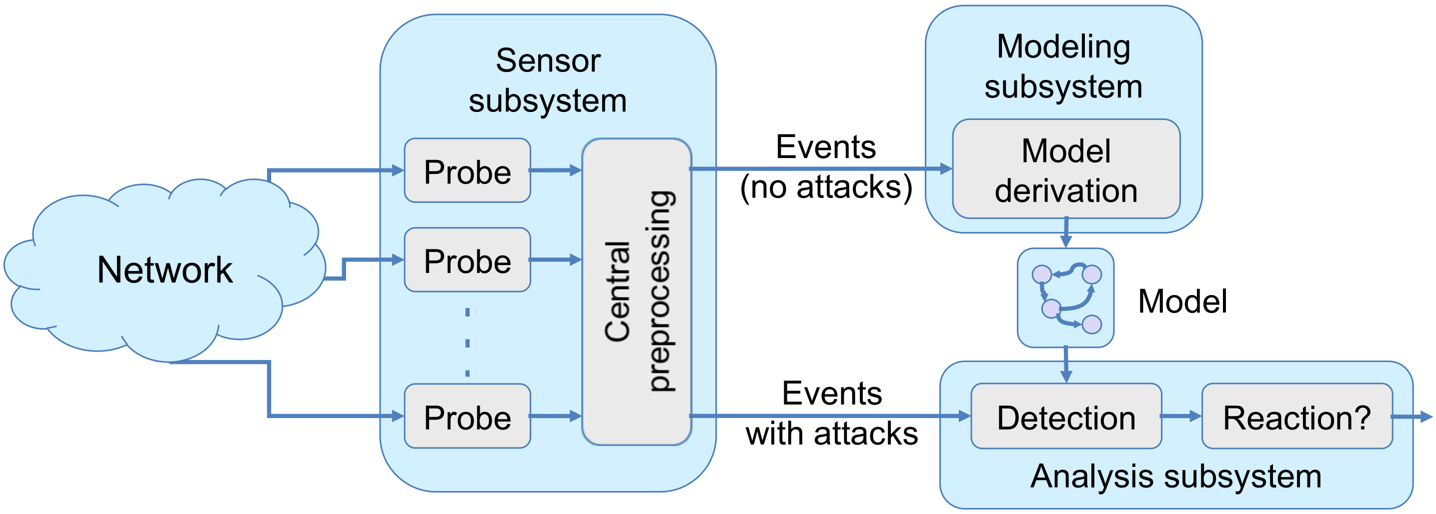
- Network abuse anomalies
  - DoS flood attacks
  - Port scans



Source: [Bar01]

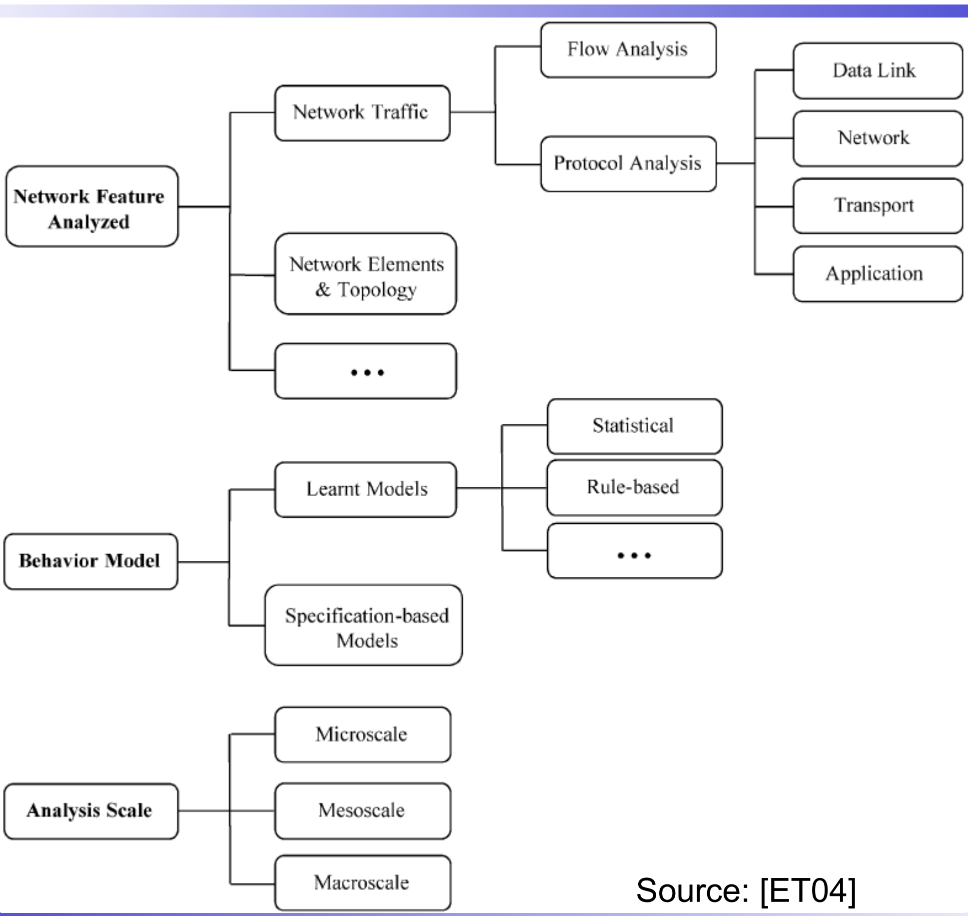
Automatic Anomaly Detection – System model

- ❑ Generic anomaly detection system



Source: [ET04]

Anomaly Detection Systems – Classification Criteria



Source: [ET04]

Automatic Anomaly Detection – Anomaly Types [CBK09]

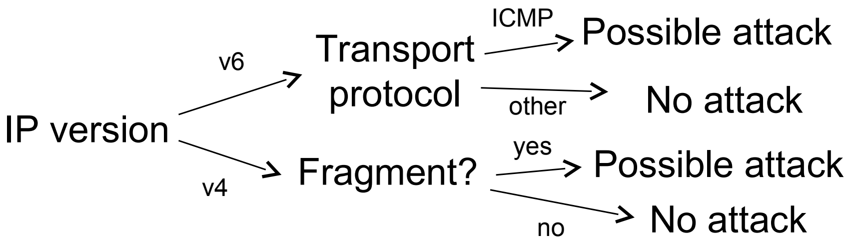
- Point Anomalies
  - Measurement points in an n-dimensional space (the lower the better → curse of dimensionality)
  - “Lonely” points or points of a small group are outliers
- Contextual Anomalies
  - Data points that are themselves not suspicious, but in their context
  - Example: Large data transfers from embedded device, low traffic at peak time
- Collective Anomalies
  - Detect deviations from a state machine
  - Data points are unsuspicious as long as they happen in a certain order
  - Deviations will be threated as an anomaly
  - Examples:
    - Retrieval of files without previously successful login (new state transition)
    - Usage of previously unused IP addresses (new state)

Automatic Anomaly Detection – Detection Methods [CBK09]

- Statistical Profiling
  - “Simple” statistical means, e.g., generating histograms, estimate parameters of distributions by maximum likelihood estimations, use regression methods to estimate curve parameters
  - Any significant change → alert
- Neural Networks
  - Neuronal networks learn normal behavior and are trained to detect attacks
  - Different designs possible, e.g., Self-Organizing Maps (SOM) to detect outliers
- Bayesian Networks
  - Method developed for artificial intelligence
  - Events are nodes in a graph, edges model dependence
  - Probabilities and dependencies are learned automatically
  - System concludes using packet information, e.g., there are only few attacks for IPv6 and few attacks use small packets → small IPv6 packets are o.k.!

Automatic Anomaly Detection – Detection Methods [CBK09]

- Support Vector Machines
  - Finding functions that separate data points caused by different machines, i.e., data points from compromised/uncompromised devices
  - Other machines also in the space of the compromised machines might also be compromised
- Rule-based Learning
  - Automatic learning of rules to sort out anomalies, e.g., decision trees
  - Example:
    - Consider there are only ICMP-based attacks for IPv6 and fragment-based attacks for IPv4
    - A decision tree would be:



Automatic Anomaly Detection – Detection Methods [CBK09]

- Clustering-based
  - Measured data points may be separated into clusters
  - If attacks are more seldom than legitimate traffic (as it should be) smaller clusters are classified to be malicious
  - Generally resource-intensive to calculate (NP-hard)
  - Popular approximation: k-Means
- Nearest-Neighbor-based
  - Simple alternative to clustering: calculate distance to closest neighbors
  - High distances indicate outliers
- Information-theory-based
  - Calculate information theoretic metrics for the normal traffic, e.g., entropy
  - When there are new traffic patterns (what could be attacks) entropy increases
  - Example: Compression of HTTP requests, if there is shell-code in it, it should be different from previous requests and less compressible

Automatic Anomaly Detection – Detection Methods [CBK09]

- Spectral analysis
  - Actually two methods
  - In time-series:
    - Derive patterns of recurring values, e.g., large file transfers once a month for backups are ok
    - E.g. using Fourier transformation
  - In graphs:
    - Reduction of dimensionality of large matrixes
    - Example: Calculation of eigenvalues in an adjacency matrix, modeling the devices communicating with each other
    - Spectral gap (difference between the two largest eigenvalues) indicates connectivity of the graph

Automatic Anomaly Detection – Example: PHAD

- Packet Header Anomaly Detection (PHAD) [Mah01]
- Old academic example, but comparably good results (back then)
- Simple protocol analysis, "learns" normal ranges of values for each header field (link, network, transport layer)
- Other values are classified anomalous

$$score_{packet} = \sum_{i \in \text{anomalous fields}} \frac{t_i n_i}{r_i}$$

t … time since previous anomaly

n … number of observations

r … number of distinct values

- Learning phase + detection phase

Automatic Anomaly Detection – Example: ALAD

- Application Layer Anomaly Detection (ALAD) [Mah02]

- Extension to PHAD, introduces conditional probabilities
- Five models:
  - P(src IP | dest IP)
    - Learns normal set of clients for each host, i.e., the set of clients allowed on a restricted service
  - P(src IP | dest IP, dest port)
    - Like (1), but one model for each server on each host
  - P(dest IP, dest port)
    - Learns the set of local servers which normally receive requests
  - P(TCP flags | dest port)
    - Learns the set of TCP flags for all packets of a particular connection
  - P(keyword | dest port)
    - Examines the text in the incoming request (first 1000 bytes)

Automatic Anomaly Detection – Properties

- Advantages:
  - Can detect unknown attacks
  - Comparably easy to setup
- Drawbacks:
  - Privacy:
    - Collecting user specific usage patterns
    - Work-related or personal habits
  - Requires continuous refreshing of normal behavior patterns
  - High number of false positives
  - Even true positives often difficult to interpret

  - If a normal behavior pattern matches an attack pattern, this kind of attack will not be detected (→ false negative)
    - What about the regular refreshes of the model?

Testing and Benchmarking of IDS

- 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: [Ath03]

Summary: Properties of the approaches

- Signature-based Detection:
  - Requires high effort in specification of rules (can be leveraged by multiple usage; comparable to sharing of virus description)
  - Effective detection of attacks that have been described in rule database
  - Unknown attacks cannot be detected
- Detection of Abnormal Behavior
  - Extremely high effort to set up
  - Possibility to detect some unknown attacks
- Anomaly Detection:
  - Theoretically challenging
  - Realization expensive in terms of required data and analysis capabilities
  - Limited Effectiveness

→ Approaches represent complementary techniques (rather than antagonistic ones)

Intrusion Prevention Systems – Motivation

- Automatic event generation nowadays not sufficient
  - Automatic exploitation is extremely fast → human intervention would be too late
  - Too many attacks on current systems → must be handled automatically for reasons of efficiency
- Led to the development of Intrusion Prevention Systems (IPS)
- Differentiation between IDS and IPS no longer meaningful as nearly all modern IDS are also IPS

Intrusion Prevention Systems – Approaches (1)

- Inline operation and suppression
  - All traffic is going through the IPS
  - Any flow (and possibly similar flows) generating an attack event will be suppressed
  - Pros:
    - Efficient
    - No race conditions
  - Cons:
    - Possible bottleneck and single point of failure
    - May be difficult to set up

Intrusion Prevention Systems – Approaches (2)

- Firewall reconfiguration
  - IPS reconfigures an existing firewall to suppress suspicious flows
  - Pros:
    - Relatively easy to set up
    - No single points of failure
  - Cons:
    - Race conditions (what if the attack already reached the target, especially if the IPS is under load?)
- Sending TCP-RST packets
  - IPS resets TCP flows by resetting the connection
  - Pros:
    - Extremely easy to setup
    - No single point of failure
  - Cons:
    - Race conditions
    - Works only for TCP

Intrusion Prevention Systems – Approaches (3)

- Deflection
  - Reconfiguration of firewall and/or routers
  - Attacker is transparently redirected to honey pots to slow down his attack
  - Pro:
    - May cause a significant slow down / confusions
  - Cons:
    - Difficult to setup (if done well)
    - Race conditions?!
- Active Defense or Automatic Hack-back
  - Academic approach (fortunately)
  - Attacks cause a manual or automatic “strike-back”
  - Used already in early 1990s by the US military to unveil “stepping stones”, i.e., proxies used by an attacker to protect his identity

Intrusion Prevention Systems – Conclusion

- Using IPS may be an option…
  - Realized approach depends on scenario
  - Not a replacement for fixing software!
- Always requires a detailed risk analysis:
  - Will the damage caused by false positives and the automatic suppression of legitimate flows, be lower than the damage prevented by suppression of illegitimate flows?
  - What about attacks from spoofed IP addresses?
- Usually only suitable for closed, well-controlled network environments…
  - E.g. preventing SQL injections in a web server

IDS Evasion

- Anomaly detection:
  - Attacker may act slowly
  - May generate high amount of “legitimate traffic” to cover attack
  - …
- Signature-based IDS: Attackers may try to construct attacks such that they are not detected
  - Works extremely well when the attacker has access to the rule set
  - May even be automated…
  - Requires countermeasures in IDS (sometimes extremely complicated)

IDS Evasion – Encoding attack vectors

- Popular methods to obfuscate attacks:
  - Recode URLs
    - Characters in URL may be expressed by different encodings
    - Example: `'a'`, `'%61'` and `'%u0061'` express all the same letter
    - Relatively easy to revert, but requires TCP reassembly
  - Recode shell code
    - Encrypt parts of the shell code (and decrypt on the fly)
    - Use different commands to achieve the same thing
    - Insert dummy commands to change the signature
    - Example: Change NOP slide from `0x90 0x90 0x90 0x90 0x90 0x90` to `0x0c0c 0x0c0c 0x0c0c` (3 times decrease register AH by 12)
    - Extremely difficult to revert

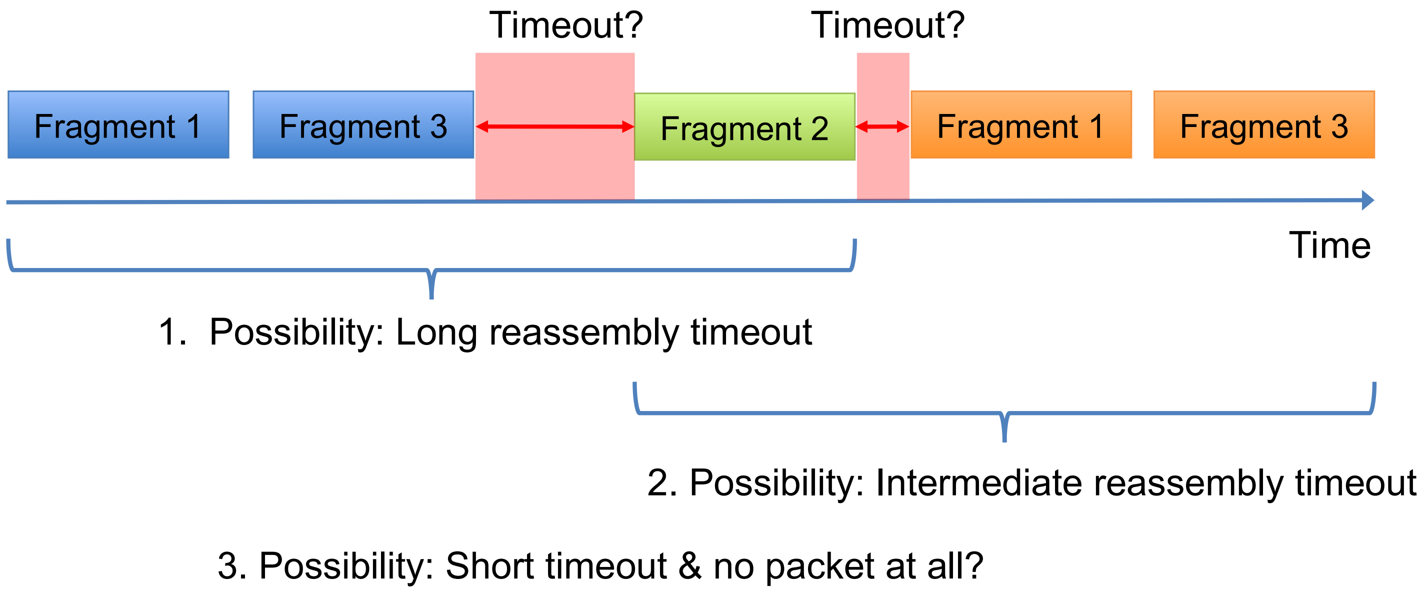
IDS Evasion – Constructing Packets

- Observation: Packet processing in IDS & end-system must be the same (otherwise different PDUs are reconstructed)
- Problem: Different OSes treat packets different as standards are ambiguous
- Examples: Overlapping TCP segments and IP fragments
  - Some OSes use first PDU part others the last send one etc.
  - IDS must either know the OS of the end-system or try all possible combinations

- Even more problematic: IDS may see packets that the end-system does not
  - Example: 1. Attacker sends (legal) TCP flow, 2. He sends a single TCP RST packet with a TTL s.t. a router behind the IDS drops it, Attacker continues TCP flow with exploit, while IDS believes in out of order packets

IDS Evasion – Considering timeouts

- Most problematic: Timeouts depend on OS & delays (especially jitter)
- Example: Timeouts for IP reassembly



- Cannot be decided securely!

General Problems of IDS (1)

- *Audit Data:*
  - Amount of log data:
    - Auditing often generates a rather high data volume
      ⇒ Significant storage capacities are required
      ⇒ Processing of audit data should be automated as much as possible
  - Location of audit data storage:
    - Alternatives: on specific “log server” or the system to be supervised
      ⇒ If stored on log server, data must be transferred to this server
      ⇒ If stored on the system to be supervised, the log uses significant amounts of resources of the system
  - Protection of audit data:
    - If a system gets compromised, audit data stored on it might get compromised either
  - Expressiveness of audit data:
    - Which information is relevant?
    - Audits often contain a rather low percentage of useful information

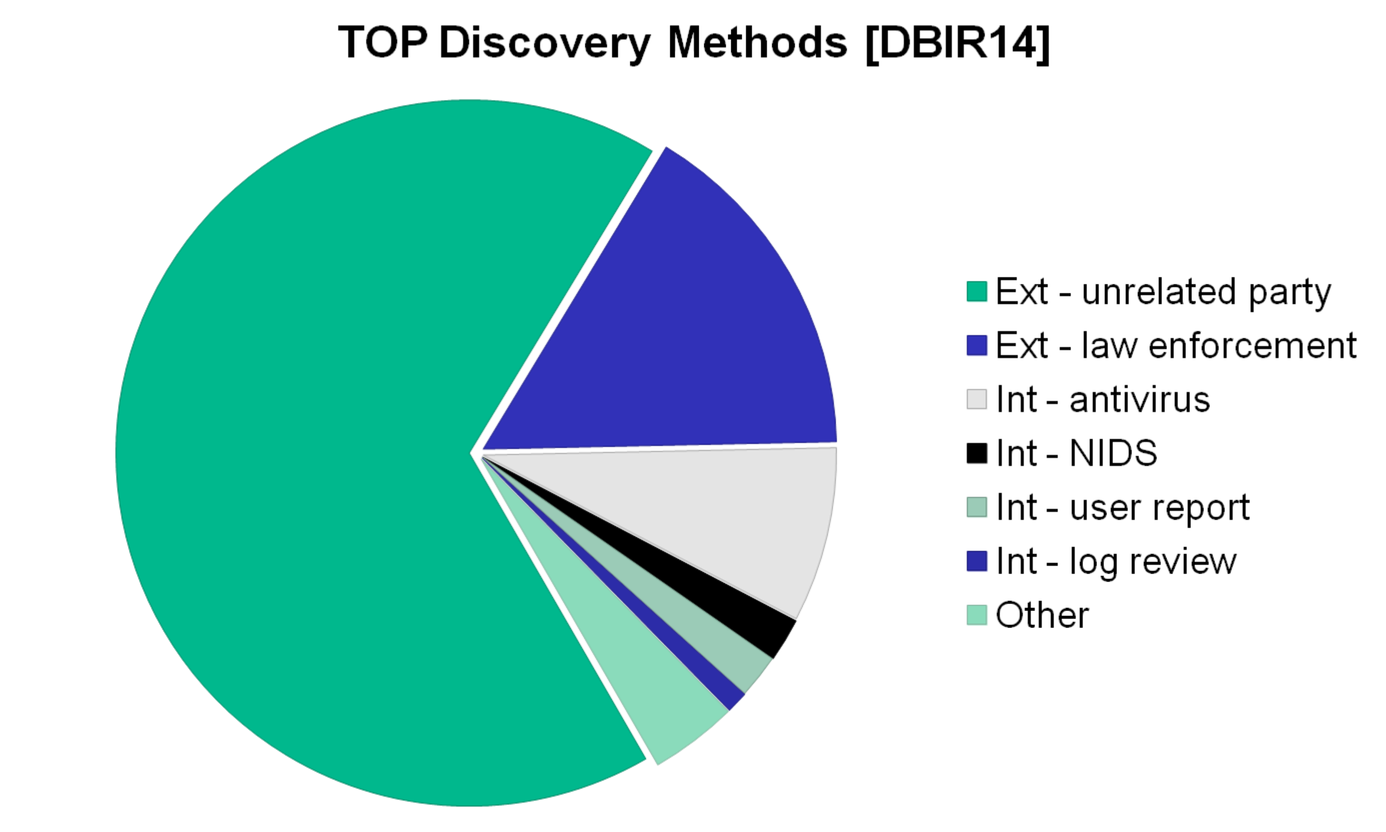
General Problems of IDS (2)

- *Privacy (→ "Datenschutz"):*
  - User identifying data elements are logged, e.g.:
    - *Directly identifying elements:* user ids
    - *Indirectly / partly identifying elements:* names of directories and subdirectories (home directory), file names, program names
    - *Minimally identifying elements:* host type + time + action, access rights + time + action
  - IDS audits may violate the privacy of users:
    - Violation of the user's right to determine himself which data is collected regarding his person
    - Collected information might be abused if not secured properly
    - Recording of events puts a psychological burden on users (→ "big brother is watching you")
  - Potential solution:
    - *Pseudonymous audit:* log activities with user pseudonyms and ensure, that they can only be mapped to user ids upon incident detection

General Problems of IDS (3)

- *Limited efficiency of analysis:*
  - Most IDS follow a centralist approach for analysis: so-called *agents* collect audit data and one central *evaluation unit* analyzes this data
    - ⇒ No (partial) evaluation in agents
    - ⇒ Performance bottleneck
  - Insufficient efficiency, especially concerning attack variants and attacks with parallel actions
- *High number of false positives:*
  - In practice, many IDS report too many false alarms (some publications report up to 10.000 per month)
  - Potential countermeasure: alarm correlation (→ hierarchical approach)
- Further problems / open issues:
  - Self protection (including strategies to cope with high load)
  - High maintenance overhead
  - Cooperation between multiple IDS

Reality check: How is cyber espionage discovered?



Additional References (1)

[Ath03] N. Athanasiades, R. Abler, J. Levine, H. Owen, G. Riley, *Intrusion Detection Testing and Benchmarking Methodologies.* Proceedings of First IEEE International Workshop on Information Assurance (IWIA'03), 2003, pp. 63.

[Bar01] P. Barford, D. Plonka, *Characteristics of Network Traffic Flow Anomalies.* Proceedings of ACM SIGCOMM Internet Measurement Workshop, October 2001.

[Bar02] P. Barford, J. Kline, D. Plonka, A. Ron, *A Signal Analysis of Network Traffic Anomalies.* Proceedings of ACM SIGCOMM Internet Measurement Workshop, Marseilles, France, November 2002.

[CBK09] V. Candola, A. Banerjee, V. Kumar. *Anomaly Detection: A Survey.* In: ACM Computing Surveys, Vol. 41, No 3, 2009.

[CDS05] G. Carle, F. Dressler, G. Schäfer. *Netzwerksicherheit - Verteilte Angriffserkennung im Internet.* Fachtagung Kommunikation in Verteilten Systemen (KiVS 2005), 28. February - 3. March 2005, Universität Kaiserslautern, Germany.

[ET04] J. M. Estevez-Tapiador, P. Garcia-Teodoro, J. E. Diaz-Verdejo. *Anomaly Detection Methods in Wired Networks: A Survey and Taxonomy.* Computer Communications, vol. 27, pp. 1569-1584, July 2004.

Additional References (2)

[Kön03] H. König. *Intrusion Detection.* Chapter XI from the lecture "Security in Computer Networks" (in German), University of Cottbus, Germany, Fall Term 2003.

[Mah01] 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.

[Mah02] M. V. Mahoney and P. K. Chan, *Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks.* Proceedings of 8th ACM International Conference on Knowledge Discovery and Data Mining, pp. 376-385, 2002.

[NN01] S. Northcutt, J. Novak. *Network Intrusion Detection - An Analyst's Handbook.* second edition, New Riders, 2001.

[VK98] VIGNA, Giovanni; KEMMERER, Richard A.: NetSTAT: A Network-based Intrusion Detection Approach. In: Proceedings of the 14th Annual Computer Security Applications Conference, S. 25–34, 1998.