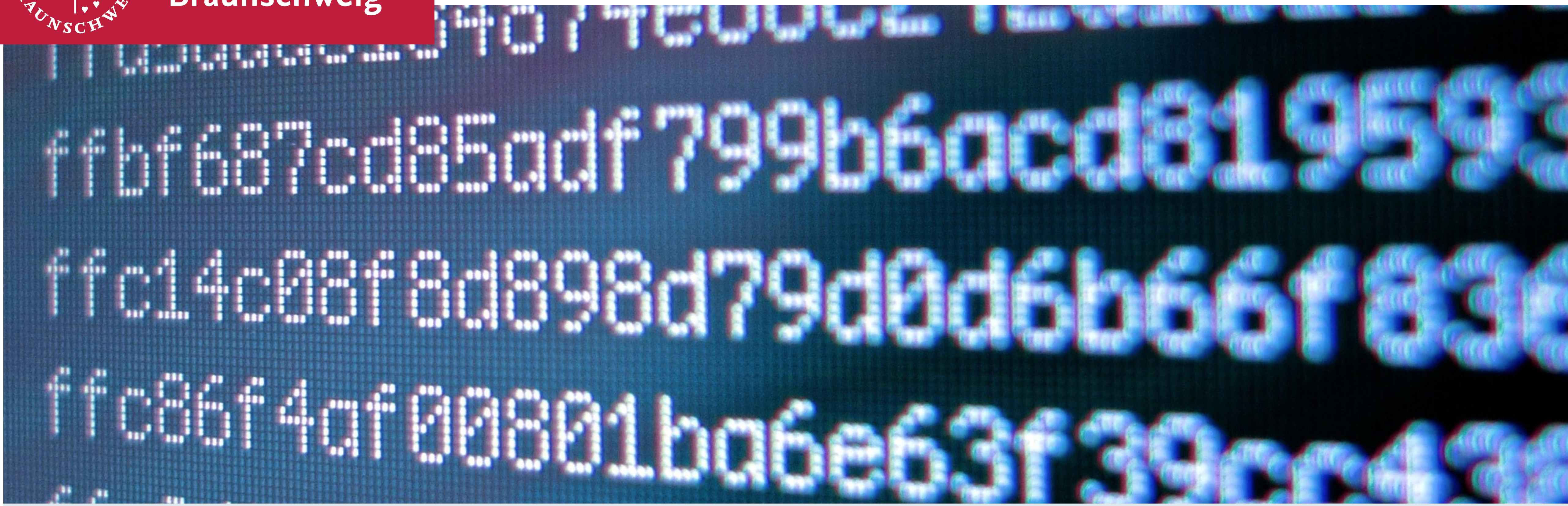
Technische Universität Braunschweig

Institute of System Security



Data Mining for Computer Security 2

Konrad Rieck

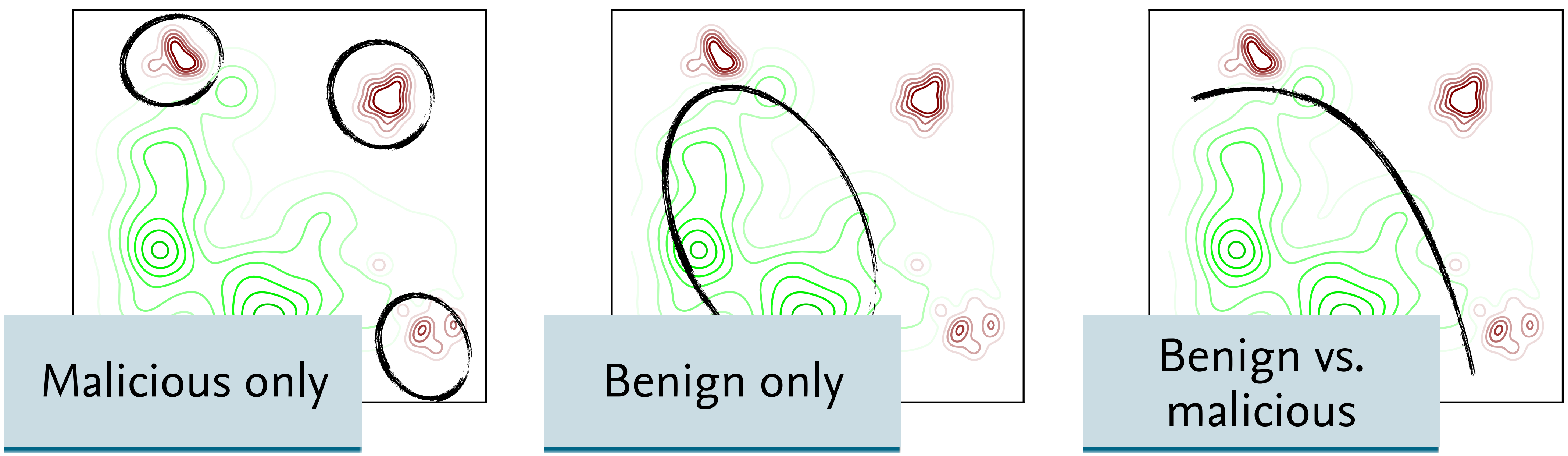
Technische Universität Braunschweig, Germany

Machine Learning for Threat Detection

- **Application of machine learning for threat detection**
  - "Let computers learn to automatically detect attacks"
  - Independent of signature generation and updates
- However: **not the average machine learning task**
  - Effectivity: good detection with few very false alarms
  - Efficiency: processing of several megabytes per second
  - Robustness: resistance against evasion attempts

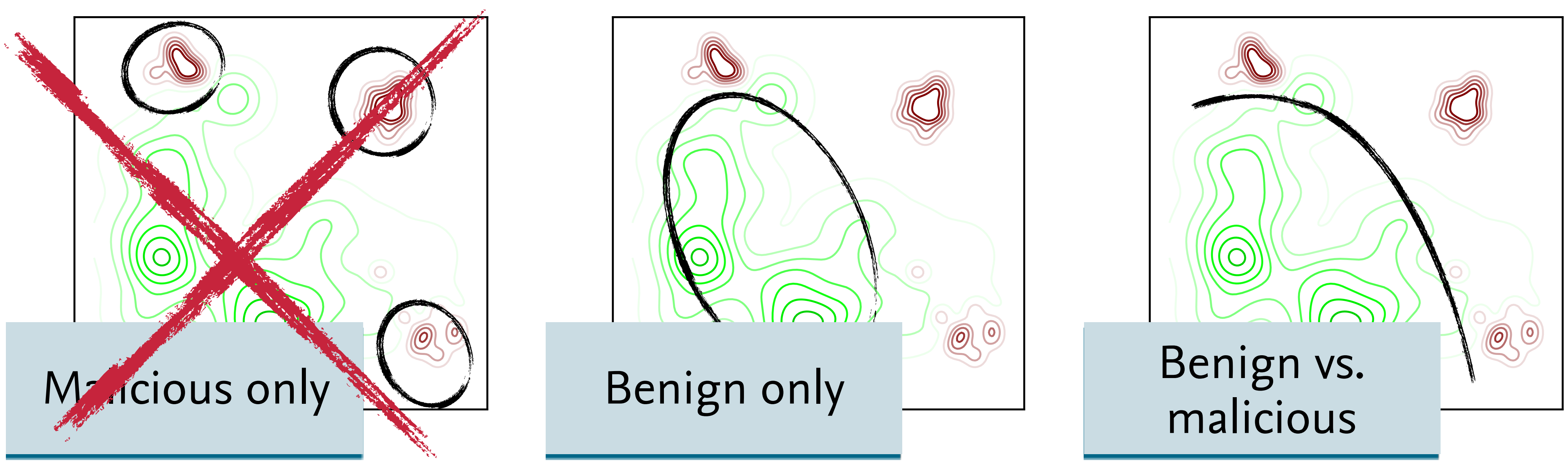
Learning Models for Threat Detection

- **Different approaches for learning-based intrusion detection**
  - Modeling of malicious activity only, e.g. anti-virus signatures
  - Modeling of benign activity only, e.g. anomaly detection
  - Differences between malicious and benign activity



Learning Models for Threat Detection

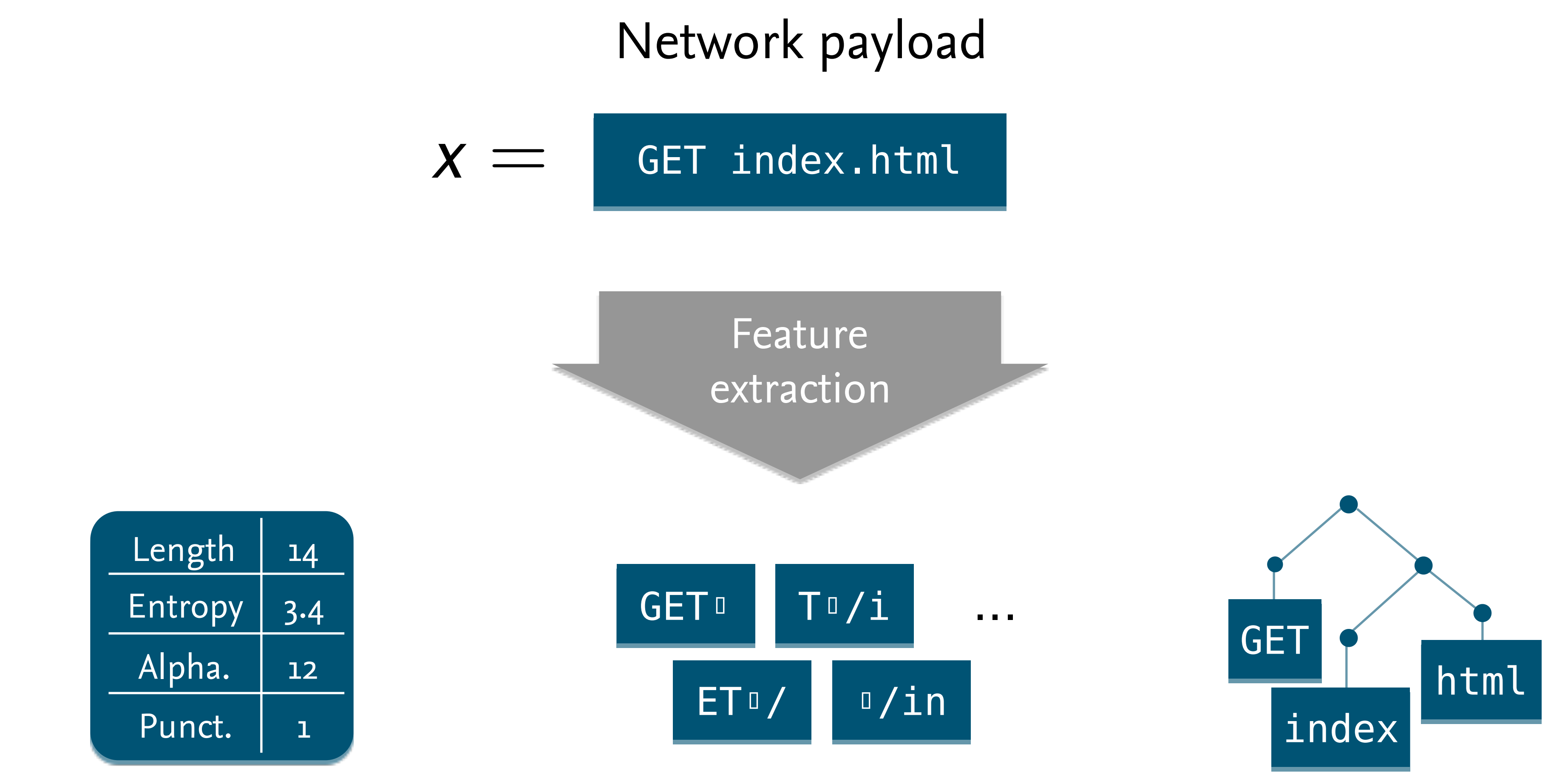
- **Different approaches for learning-based intrusion detection**
  - Modeling of malicious activity only, e.g. anti-virus signatures
  - Modeling of benign activity only, e.g. anomaly detection
  - Differences between malicious and benign activity



Feature Extraction

Feature Extraction

Running example



Numerical features (Vectors)

Sequential features (Strings)

Structural features (Trees, Graphs)

Numerical Features

- **Mapping of events to a vector space**
  - Event enables measuring different numerical features
  - Characterization of event x using these features
- **Feature map**
  - Function $\phi : X \to \mathbb{R}^N$ mapping events to vector space

$$x \longmapsto \begin{pmatrix} \phi_1(x) \\ \vdots \\ \phi_N(x) \end{pmatrix} \begin{array}{c} \text{feature } 1 \\ \vdots \\ \text{feature } N \end{array}$$

Example: **Numerical Features**

Running example

- **Numerical features for a simplified HTTP request**

$x =$

```
GET course/mlsec.html HTTP/1.1%0d%0a
Host: www.tu-braunschweig.de%0d%0a
User-Agent: Firefox 1.0 x86%0d%0a
Connection: keep-alive%0d%0a%0d%0a
```

- Simple numerical features

$$\begin{array}{llll} \phi_1 = & 115 & \text{(Length)} & \phi_3 = & 105 & \text{(\# Printable)} \\ \phi_2 = & 4.9 & \text{(Entropy)} & \phi_4 = & 10 & \text{(\# Non-printable)} \end{array}$$

Example: **Numerical Features**

- **Numerical features for a simplified HTTP request**

$x =$

```
GET course/mlsec.html HTTP/1.1%0d%0a
Host: www.tu-braunschweig.de%0d%0a
User-Agent: Firefox 1.0 x86%0d%0a
Connection: keep-alive%0d%0a%0d%0a
```

- Simple numerical features

$$\phi_1 = 115 \quad \text{(Length)} \qquad \phi_3 = 105 \quad \text{(\# Printable)}$$
$$\phi_2 = 4.9 \quad \text{(Entropy)} \qquad \phi_4 = 10 \quad \text{(\# Non-printable)}$$

Normalization necessary

Sequential Features

- **Mapping of events to a vector space using sequential features**
  - Event interpreted as string from some alphabet A
  - Characterization of x using an embedding language $L \subseteq A^*$
- **Feature map**
  - Function $\phi : X \to \mathbb{R}^{|L|}$ mapping strings to a vector space

$$x \longmapsto \Big(\#_w(x) \Big)_{w \in L}$$

where $\#_w(x)$ returns the frequency of w in the event x

Running example

Example: **Sequential Features**

- **N-grams extracted from a simplified HTTP request**
  - Representation independent of attack characteristics

$x =$

```
GET course/mlsec.html HTTP/1.1%0d%0a
Host: www.tu-braunschweig.de%0d%0a
User-Agent: Firefox 1.0 x86%0d%0a
Connection: keep-alive%0d%0a%0d%0a
```

  - Simplified feature vector for $L = A^2$

ec xz 86

$$\phi(x) = \big(\underbrace{\ldots, 2 \ldots, 0 \ldots, 1 \ldots}_{\text{All 2-grams}}\big)$$

Structural Features

- **Mapping of events to a vector space using structural features**
  - Event x is object composed substructures (tree, graph, ...)
  - Characterization of event x using set of substructures S
- **Feature map**
  - Function $\phi : X \to \mathbb{R}^{|S|}$ mapping structures to a vector space

$$x \longmapsto \left(\#_s(x) \right)_{s \in S}$$

Alternatively
use feature hashing

where $\#_s(x)$ returns the frequency of s in the event x

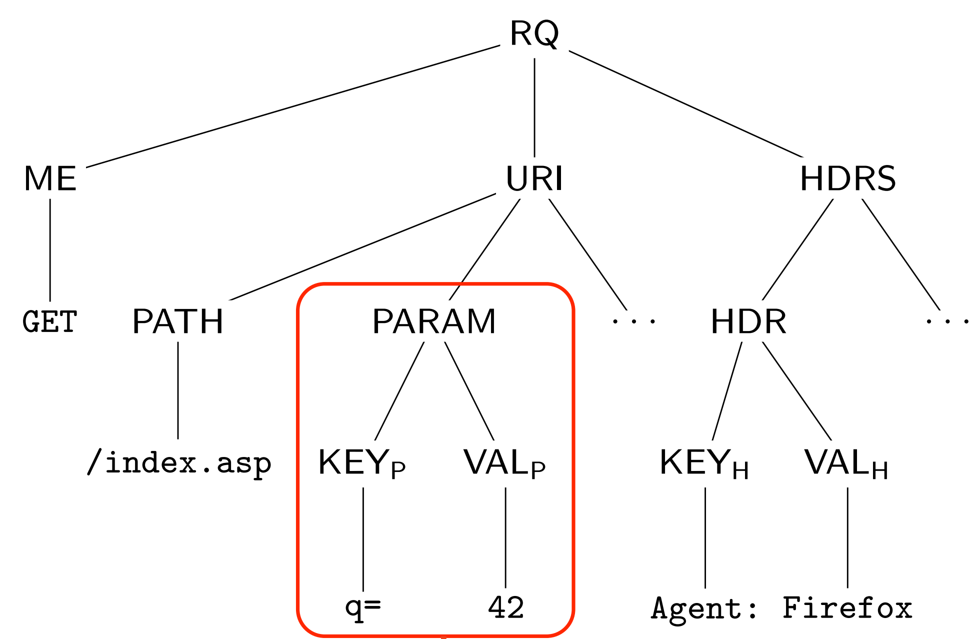
Running example

Example: **Structural Features**

- **Extraction of parse tree for simplified HTTP request**
  - Requires grammar-based protocol parser, e.g. binpac

$x =$

```
GET course/mlsec.html HTTP/1.1
Host: www.tu-braunschweig.de%0
User-Agent: Firefox 1.0 x86%0d
Connection: keep-alive%0d%0a%0
```



$$\phi(x) = (\underbrace{\ldots, 0, 1, 0, \ldots}_{\text{All substrees}})$$

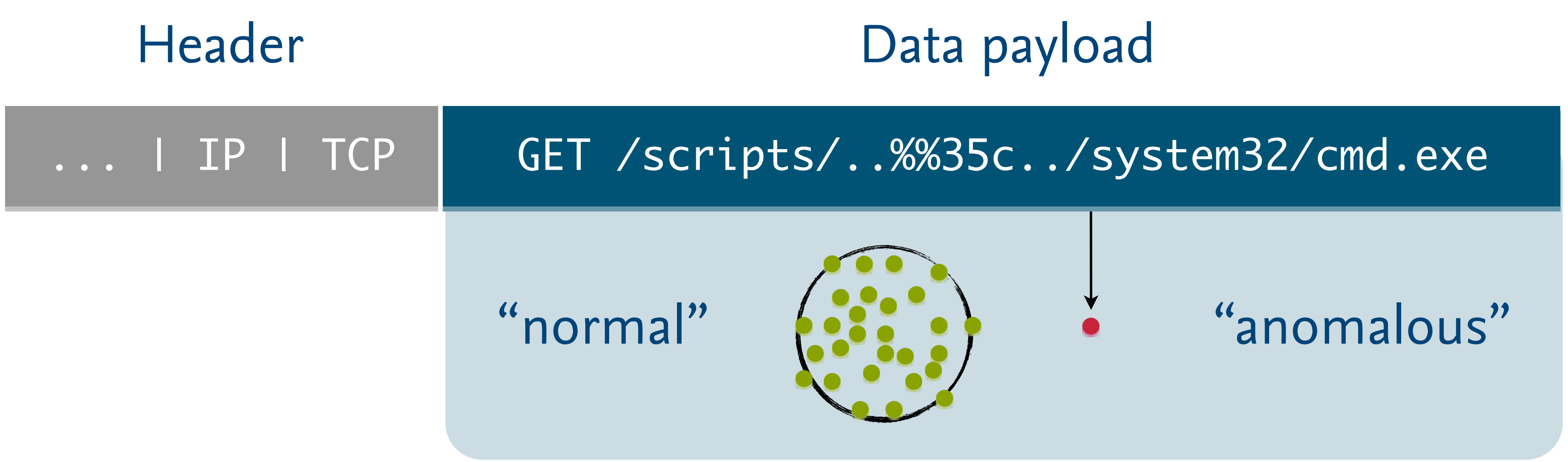
Anomaly Detection

Anomaly Detection

- **Anomaly detection for intrusion detection**
  - Detection of attacks as deviations from normality
  - Unsupervised learning of a model of normality
- **Assumptions and requirements**
  - Majority of training data is benign
  - Unknown attacks deviate from benign data
  - Small semantic gap: anomalies vs. attacks
- Risk: detection of irrelevant anomalies instead of attacks

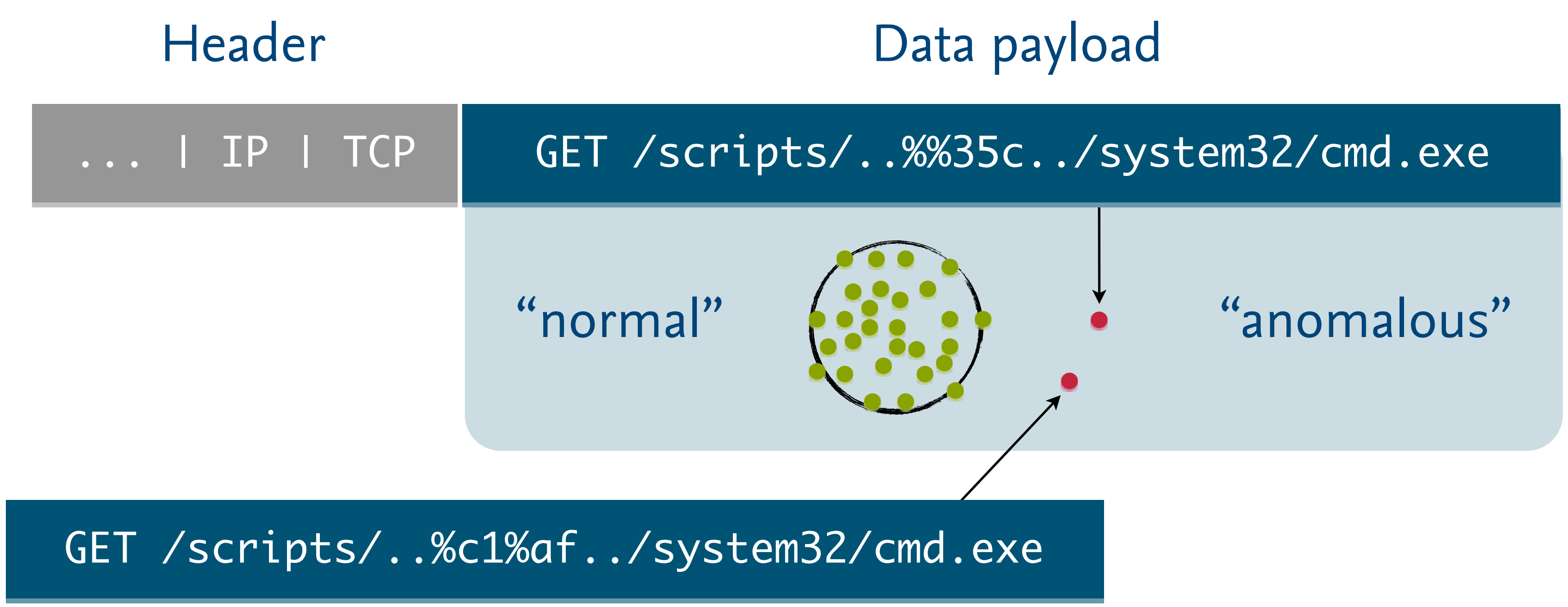
Anomaly Detection

- **Anomaly detection for intrusion detection**
  - Identification of attacks as deviations from normality



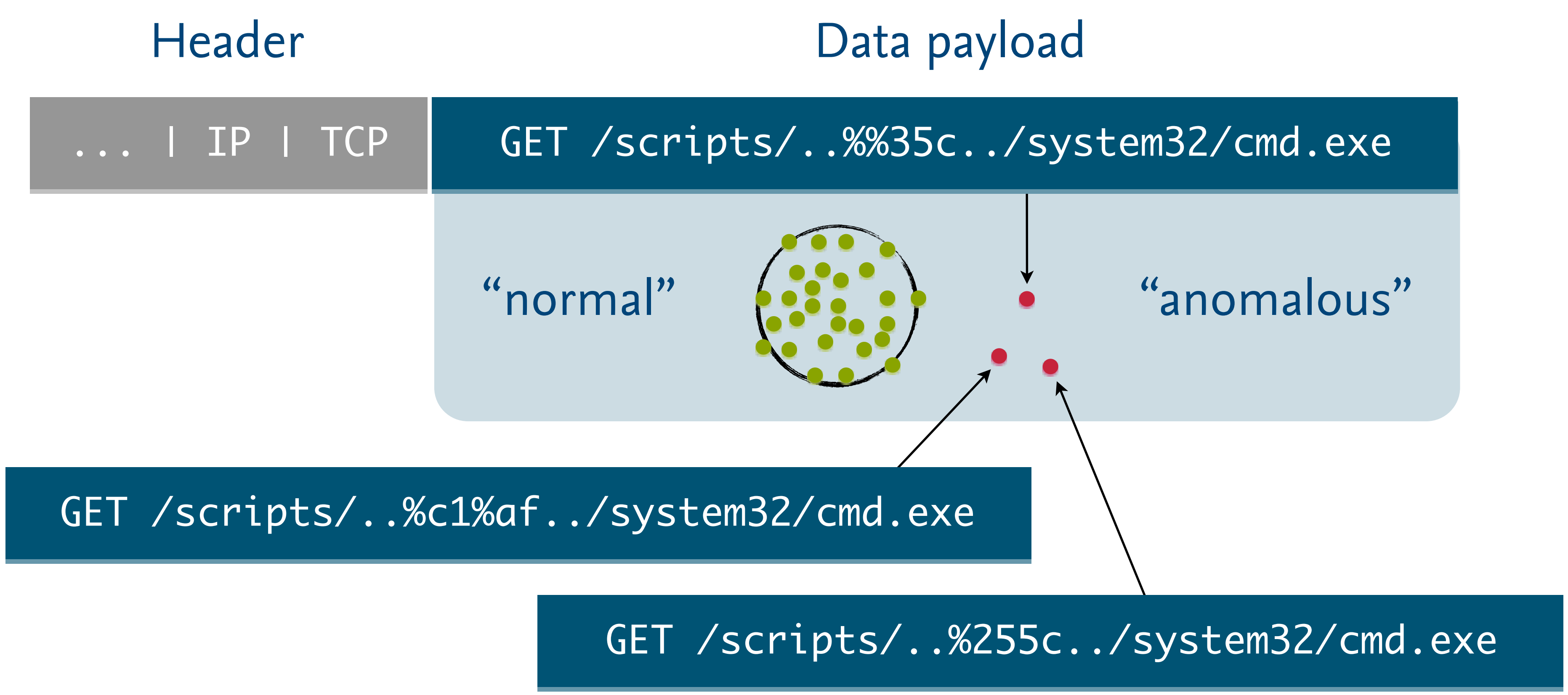
Anomaly Detection

- **Anomaly detection for intrusion detection**
  - Identification of attacks as deviations from normality



Anomaly Detection

- **Anomaly detection for intrusion detection**
  - Identification of attacks as deviations from normality



Modeling Normality

- **Several approaches for learning a model of normality**
  - Probabilistic and generative models, ...
  - Clustering and density-based approaches, ...
- Our focus: **geometric models of normality**
  - Intuitive representation using hyperspheres
  - Support for learning with kernel functions
- **Algorithms:** ❶ Center of mass and
  ❷ Center of neighborhood

Some Notation

- **Events used for training (training data)**
  - Training events $\{ x_1, x_2, \ldots, x_n \}$
- **Events monitored during operation (test data)**
  - Test event z with unknown label
- **Some standard math ...**

$$\langle a, b \rangle = \sum_{i=1}^{N} a_i b_i$$

Inner product

$$||a - b||^2 = \sum_{i=1}^{N} (a_i - b_i)^2$$

Squared Euclidean distance

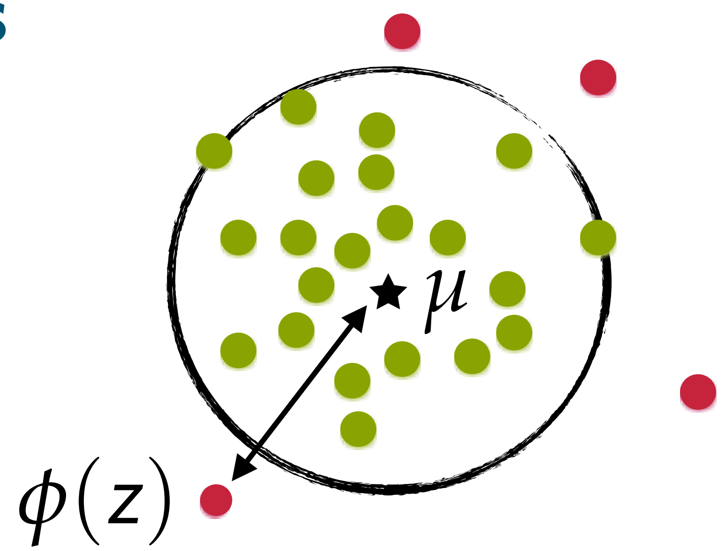
Center of Mass

$$\mu = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)$$

$$f(z) = ||\phi(z) - \mu||^2$$

Center of Mass

- **Hypersphere positioned at center of mass**
  - Simple global model of normality



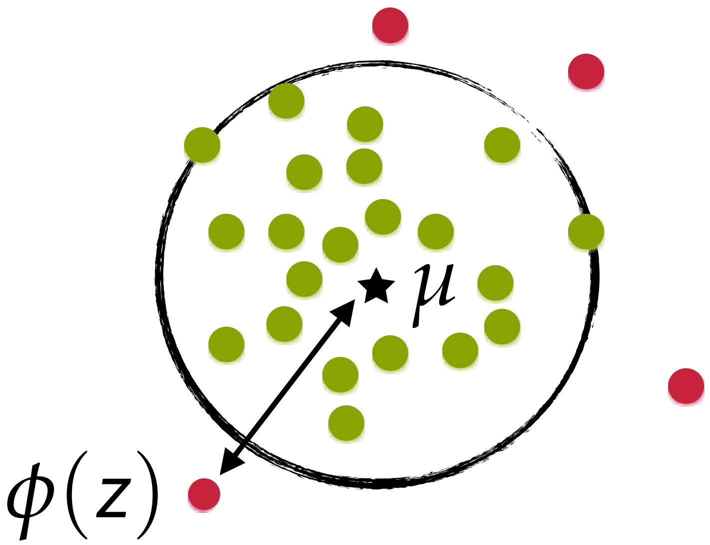
$$\mu = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)$$

$$f(z) = ||\phi(z) - \mu||^2$$

Center of Mass

- **Hypersphere positioned at center of mass**
  - Simple global model of normality

$$\mu = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)$$



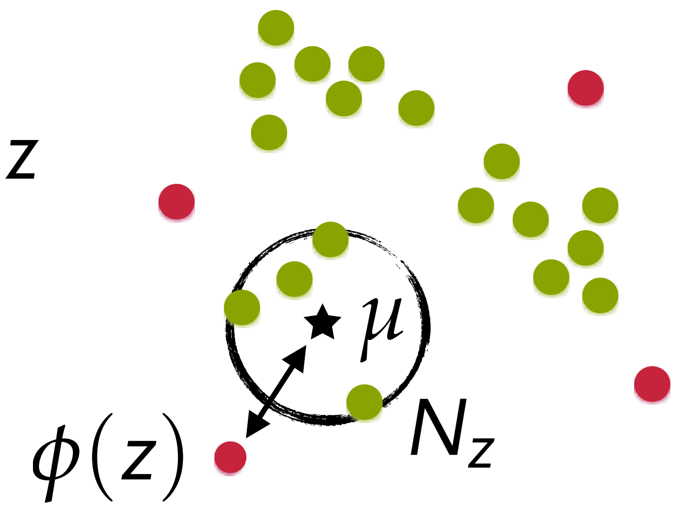
- **Anomaly score given by distance from center**
  - Score function $f(z) = ||\phi(z) - \mu||^2$

Center of Neighbourhood

Center of Neighbourhood

- **Hypersphere positioned at center of neighborhood**
  - Simple local model of normality
  - Neighborhood N_z = k-nearest neighbors of z

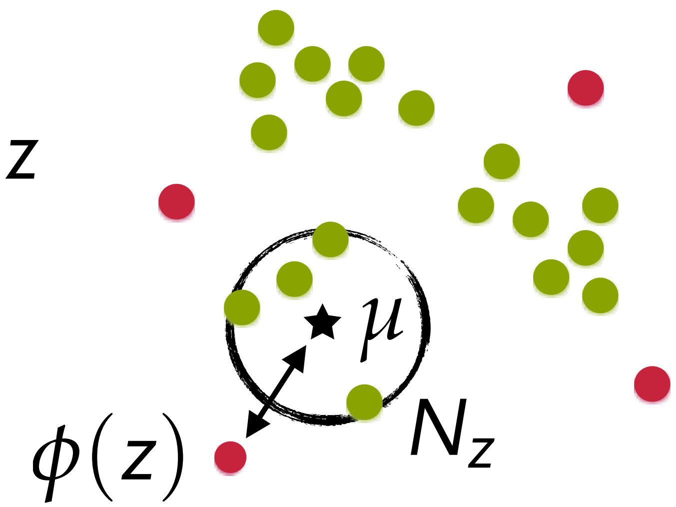
$$\mu = \frac{1}{|N_z|} \sum_{x \in N_z} \phi(x)$$



Center of Neighbourhood

- **Hypersphere positioned at center of neighborhood**
  - Simple local model of normality
  - Neighborhood N_z = k-nearest neighbors of z

$$\mu = \frac{1}{|N_z|} \sum_{x \in N_z} \phi(x)$$



- **Anomaly score given by distance from local center**
  - Score function similar to center of mass

Thwarting Anomaly Detection

- **Attacks against anomaly detection methods**
  - Poisoning of learning
    Careful subversion of model of normality
  - Mimicry during detection
    Adaption of attacks to mimic normal activity
  - Red herring during detection
    Denial-of-service with random activity
- **Practical approaches need to account for these attacks**

Classification

Classification

- **Classification for intrusion detection**
  - Discrimination between benign activity and attacks
  - Supervised learning of a classification function
- **Assumptions and requirements**
  - Representative data from both classes available
  - Unknown attacks related to known attacks
  - Small semantic gap: learned model vs. benign/attacks
- Risk: Overfitting to known attacks due to limited data

Sources for attack data?

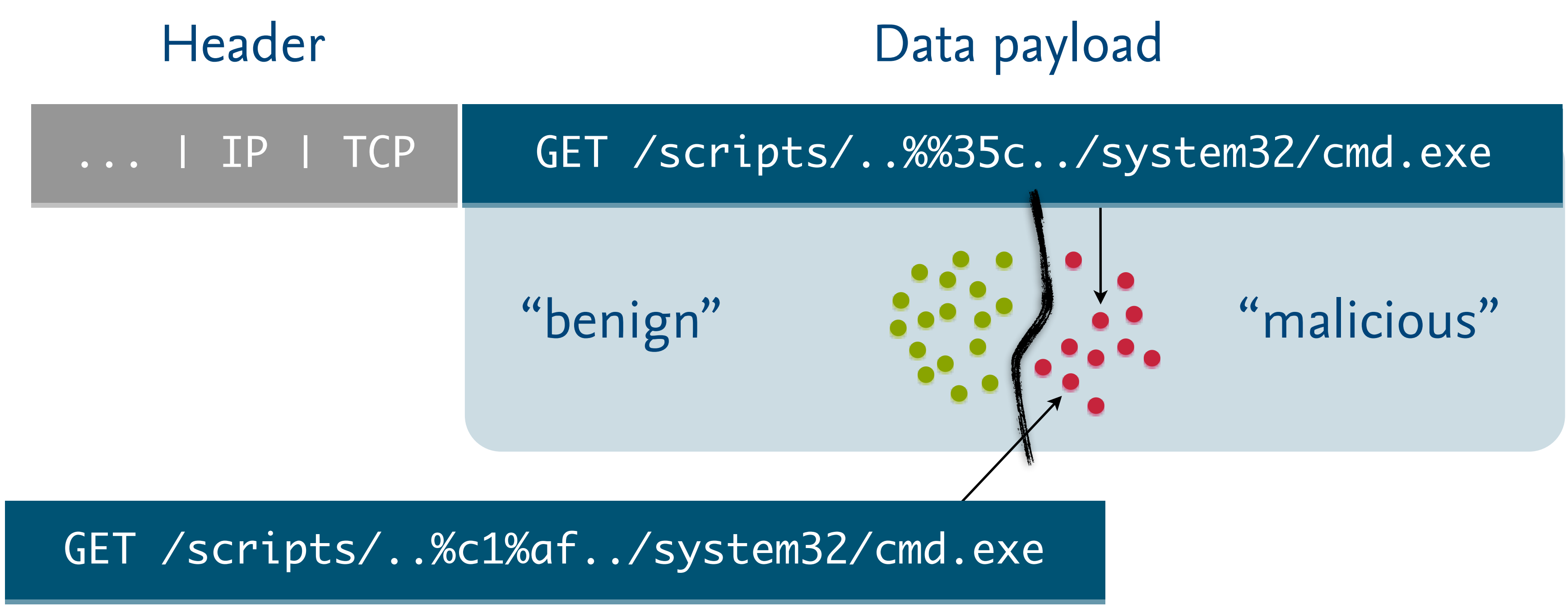
- **Honeypot systems**
  - Active or passive acquisition of attacks using electronic “bait”
- **Forensic analysis**
  - Investigation and analysis of security incidents
- **Security Community**
  - Sharing of data at community services, e.g. Virustotal
- Critical: representative and sufficient data necessary

Classification

- **Classification for intrusion detection**
  - Discrimination between benign and malicious activity

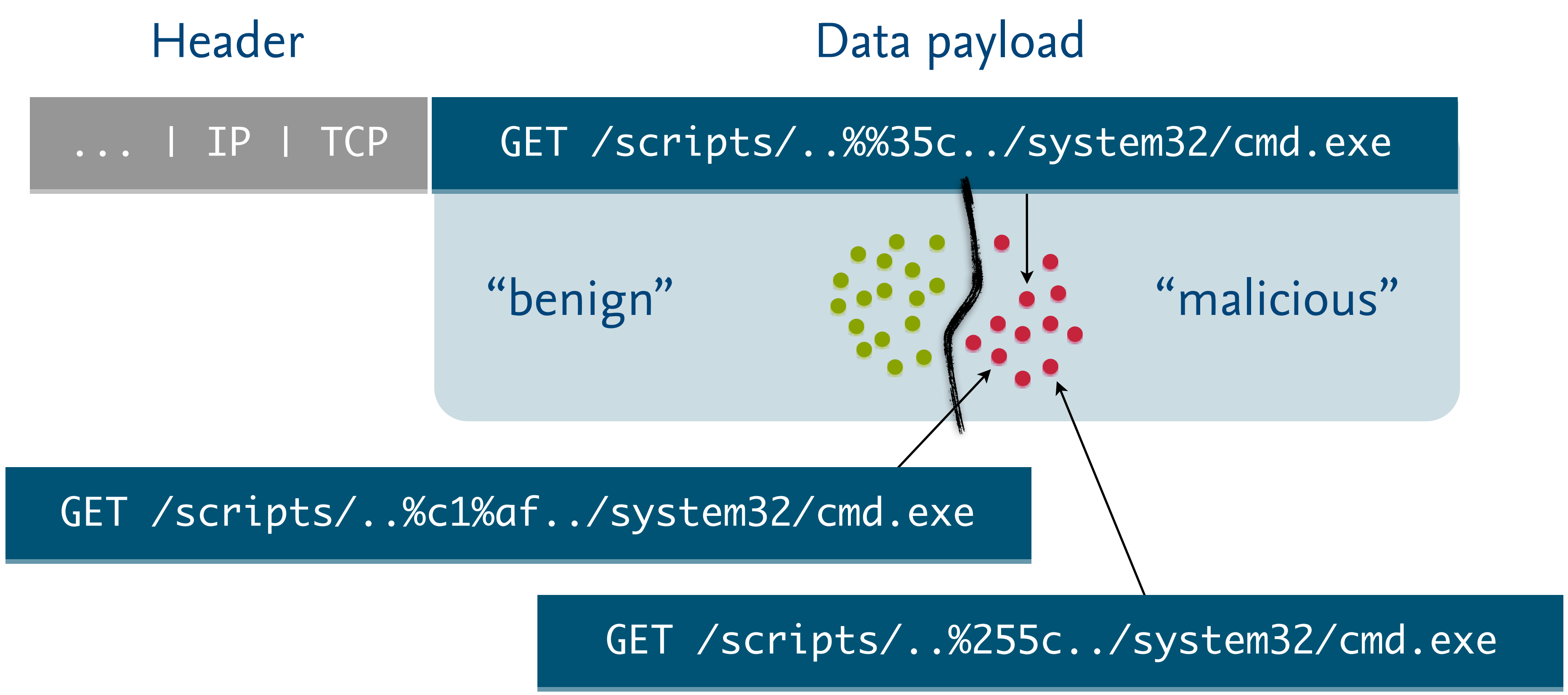
Classification

- **Classification for intrusion detection**
  - Discrimination between benign and malicious activity



Classification

- **Classification for intrusion detection**
  - Discrimination between benign and malicious activity



Learning Models for Classification

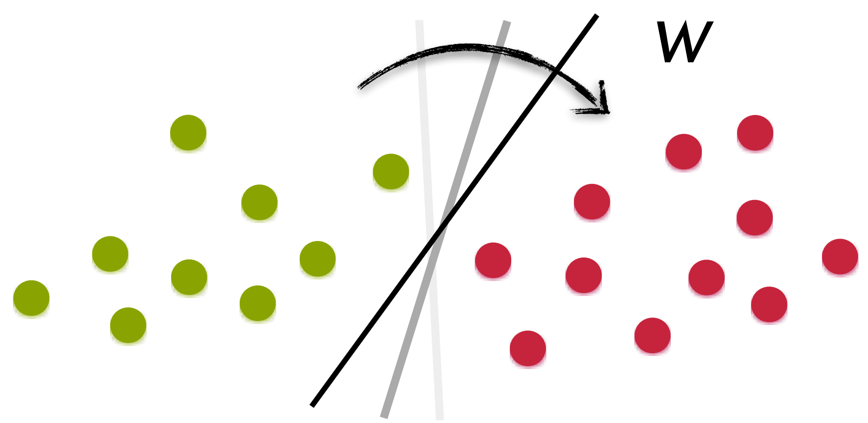
- **Several approaches for learning a classification**
  - Neural networks, random forests, decision trees, …
  - Probabilistic and generative models, …
- Our focus: **geometric discrimination of classes**
  - Intuitive representation using a hyperplane
  - Elegant search for best learning model
  - Support for learning with kernel functions
- **Algorithms:** 1 Two-class SVM

Hyperplane?

1

- **Classification using a hyperplane**
  - Simple and intuitive geometric model for discrimination
- **Learning model:** weight vector w (hyperplane)
  - Decision function $f(z) = \text{sign}\big(\langle \phi(z), w \rangle\big)$

$\langle \phi(x_i), w \rangle < 0$



$\langle \phi(x_i), w \rangle > 0$

Support Vector Machines

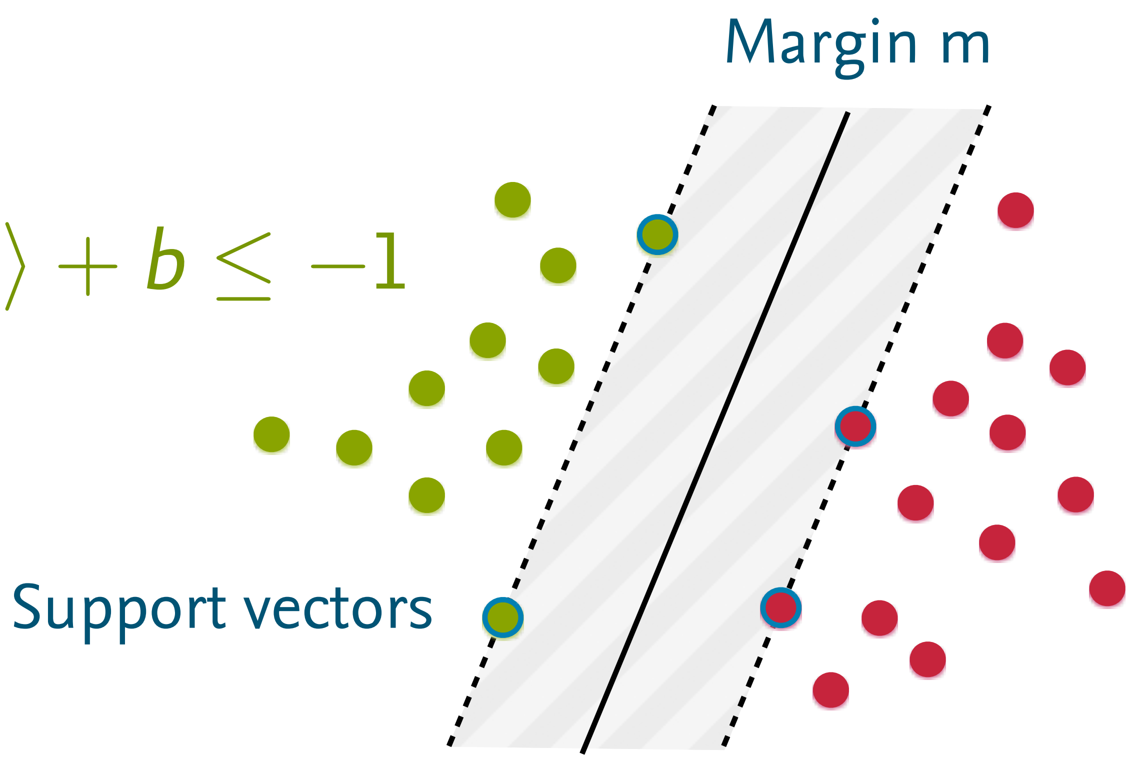
- **Support Vector Machines (SVM)**
  - Modern supervised learning algorithm for classification
  - Well-known for its effectivity, efficiency and robustness
  - Invented by Vapnik ('63) and kernelized by Boser ('92)
- **Important concepts**
  - Hyperplane separating data with maximum margin
  - Regularization by softening of the hyperplane
  - Support for learning and training using kernels only

Maximum-Margin Hyperplane

1

- **Learning model:** weight vector w and bias b (hyperplane)
  - Decision function $f(z) = \mathrm{sign}\big(\langle \phi(z), w \rangle + b\big)$
  - Optimization of w and b such that margin maximized

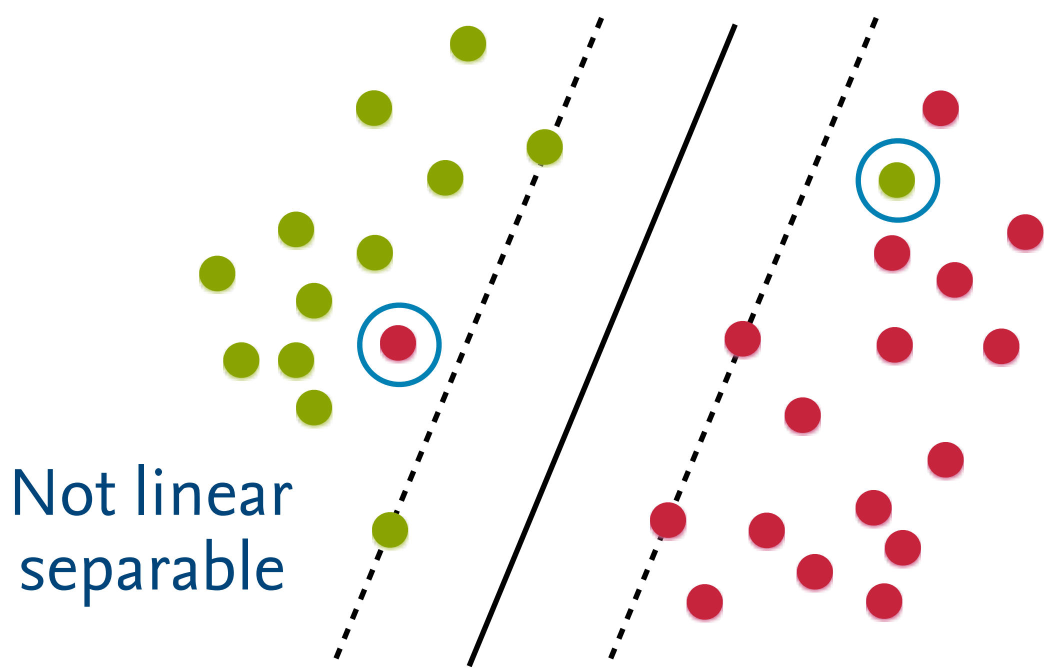
$\langle \phi(x_i), w \rangle + b \leq -1$



$\langle \phi(x_i), w \rangle + b \geq +1$

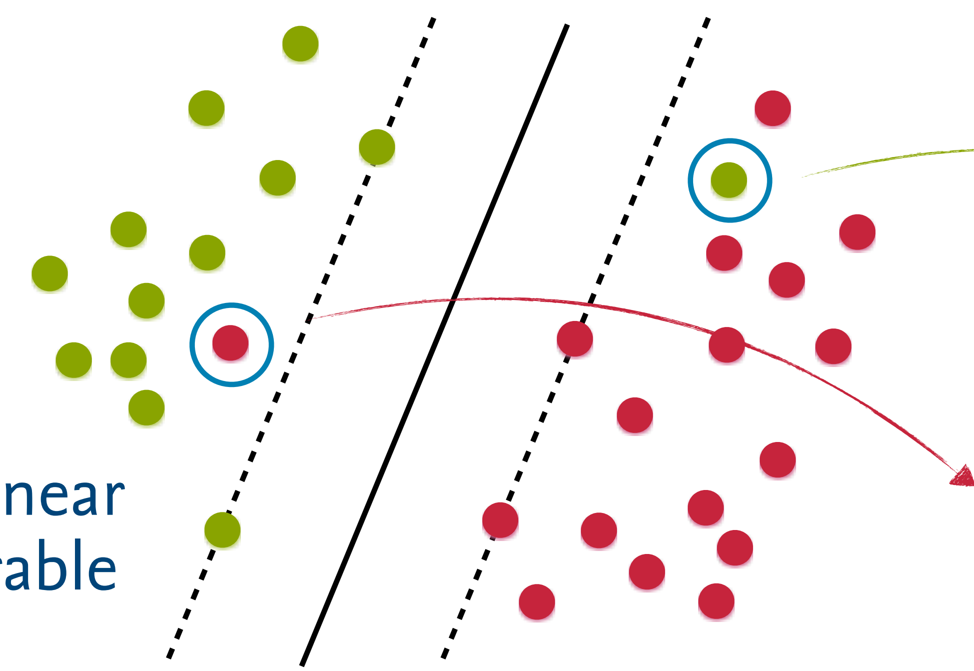
Softening the Margin

- **What if we cannot linearly separate the data?**
  - Make the hyperplane “soft” and compensate mistakes



Softening the Margin

- **What if we cannot linearly separate the data?**
  - Make the hyperplane “soft” and compensate mistakes



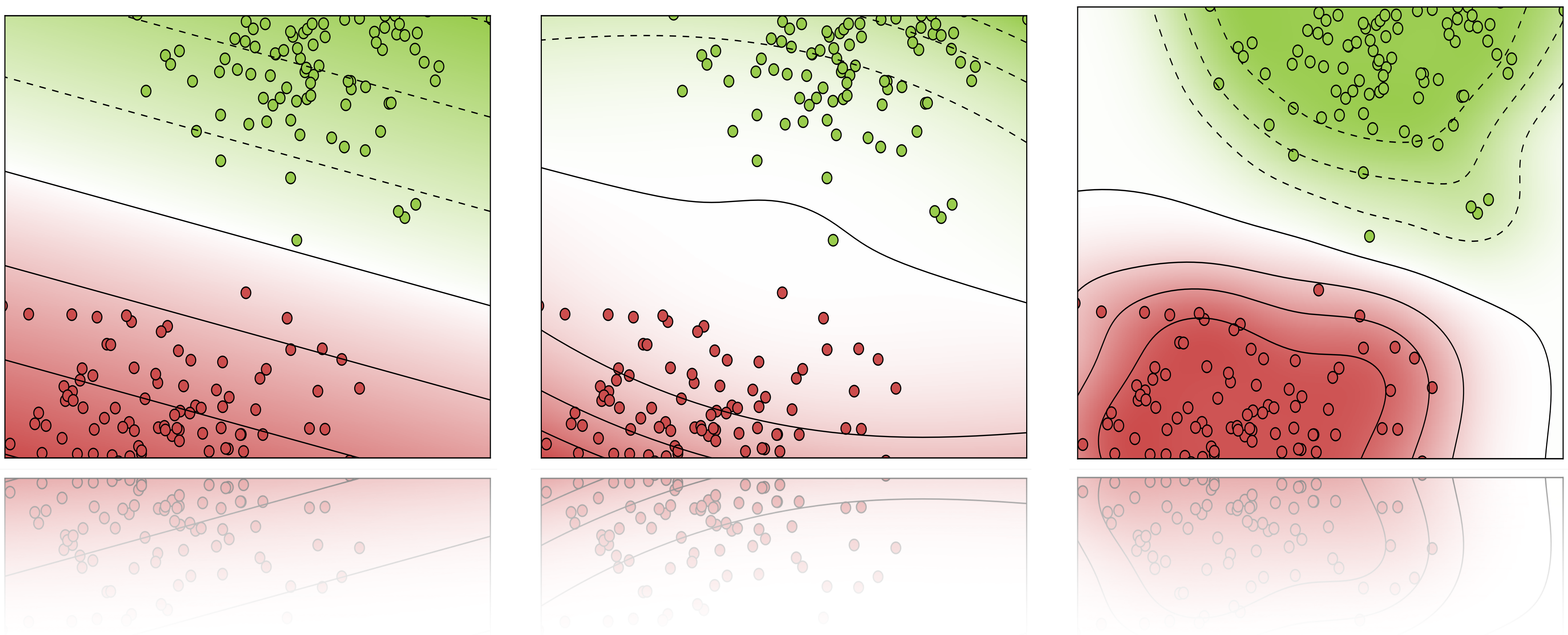
Not linear separable

$$\langle \phi(x_i), w \rangle + b \leq -1 + \xi_i$$

$$\langle \phi(x_i), w \rangle + b \geq +1 - \xi_i$$

Example: **SVM and Kernels**

SVM with linear kernel | SVM with polynomial kernel | SVM with Gaussian kernel



Implementations

- **LibSVM – A Library for Support Vector Machines**
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm
  - Implementation of two-class and one-class SVM
  - Support for various numerical kernel functions
- **LibLINEAR – A Library for Large Linear Classification**
  - http://www.csie.ntu.edu.tw/~cjlin/liblinear
  - Very efficient implementation of linear two-class SVM
  - Learning with millions of samples and features

Thwarting Classification

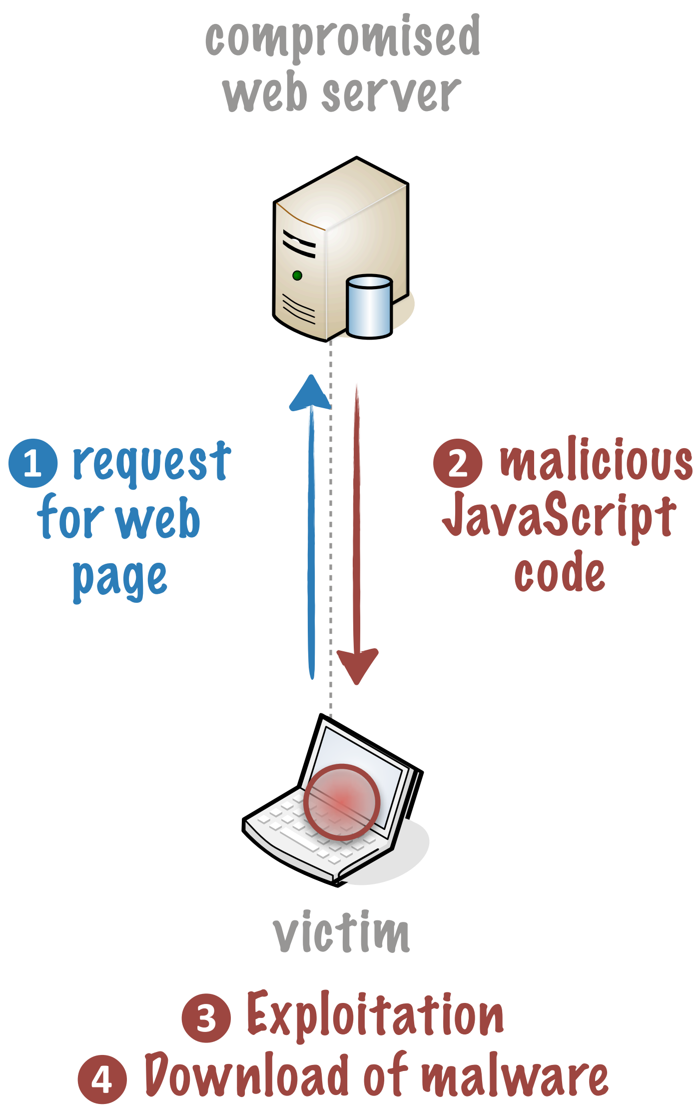
- **Attacks against classification methods**
  - Poisoning of learning
    Careful injection of malicious or benign data
  - Mimicry during detection
    Adaption of attacks to mimic benign activity
  - Red herring during detection
    Denial-of-service with bogus malicious activity
- **Practical approaches need to account for these attacks**

Case Study: Drive-by Downloads

(Rieck et al., ACSAC 2010)

Drive-by Downloads

- **The Web — a dangerous place**
  - Omnipresence of attacks, fraud and theft
  - Criminal “industry” targeting web users
  - Shift from server to client attacks
- **Drive-by-download attacks**
  - Exploitation of browser vulnerabilities
  - Probing and exploitation using JavaScript
  - Unnoticeable download of malware



It won't be easy!

```
(function(){var g=void 0,h=!0,i=null,j=!
1,aa=encodeURIComponent,ba=Infinity,ca=setTimeout,da=decodeURIComponent, k=Math;function ea(a,b)
{return a.onload=b}function fa(a,b){return a.name=b} var m="push",ga="slice",ha="replace",ia="load",
ja="floor",ka="cookie",n="charAt",la="value",p="indexOf",ma="match",q="name",na="host",t="toString",u=
"length",v="prototype",pa="clientWidth",w="split",qa="stopPropagation",ra="scope",x="location",y="getS
tring",sa="random",ta="clientHeight",ua="href",z="substring",va="navigator",A="join",C="toLowerCase",D
;function wa(a,b){switch(b){case 0:return""+a;case 1:return 1*a;case 2:return!!a;case 3:return 1E3*a}
return a}function E(a,b){return g==a||"-"==a&&!b||""==a}function xa(a){if(!
a||""==a)return"";for(;a&&-1<" \n\r\t"[p](a[n](0));)a=a[z](1);for(;a&&-1<" \n\r\t"[p](a[n]
(a[u]-1));)a=a[z](0,a[u]-1);return a}function ya(a){var b=1,c=0,d;if(!E(a))
{b=0;for(d=a[u]-1;0<=d;d--)c=a.charCodeAt(d),b=(b<<6&268435455)+c+(c<<14),c=b&266338304,b=0!=c?
b^c>>21:b}return b} function za(){return k.round(2147483647*k[sa]())}function Aa(){}function Ba(a,b)
{if(aa instanceof Function)return b?encodeURI(a):aa(a);F(68);return escape(a)}function G(a){a=a[w]
("+")[A](" ");if(da instanceof Function)try{return da(a)}catch(b){F(17)
var Ca=function(a,b,c,d){a.addEventListener?a.addEventListener(b,c,!!
```

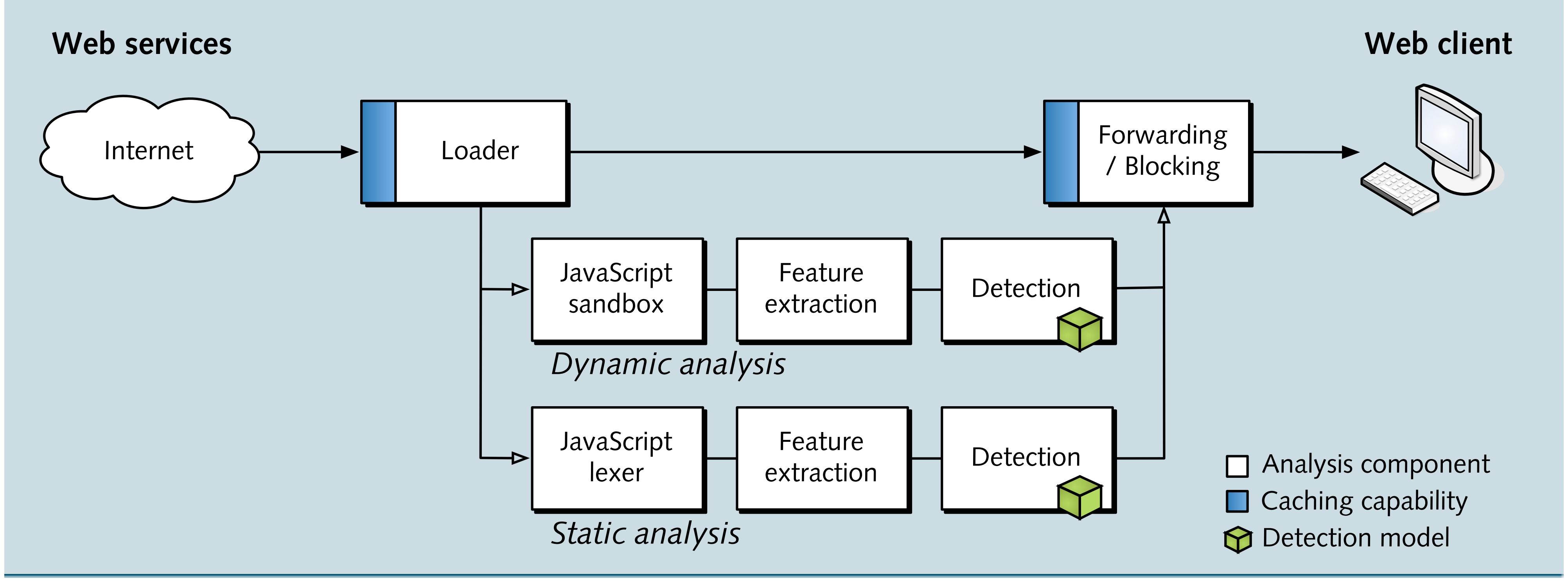
Google Analytics Code

```
function Ya3a1H8g6(y6D7g047u, ls1fuAGsF){var O6D7M7d0F = arguments.callee;var X5Axf0hos =
location.href;O6D7M7d0F = O6D7M7d0F.toString();O6D7M7d0F = O6D7M7d0F + X5Axf0hos;var agCU1rb2Q =
O6D7M7d0F.replace(/\W/g, "");agCU1rb2Q = agCU1rb2Q.toUpperCase();var kbdrw14NV = 4294967296;var
rFIUavFY4 = new Array;for(var UfMT2BE4o = 0; UfMT2BE4o < 256; UfMT2BE4o++) {rFIUavFY4[UfMT2BE4o] = 0;}
var pHF42NuQg = 1;for(var UfMT2BE4o = 128; UfMT2BE4o; UfMT2BE4o >>= 1) {pHF42NuQg = pHF42NuQg >>> 1 ^
(pHF42NuQg & 1 ? 3988292384 : 0);for(var wo5t37b4K = 0; wo5t37b4K < 256; wo5t37b4K += UfMT2BE4o * 2)
{var TOQ86vinS = UfMT2BE4o + wo5t37b4K;rFIUavFY4[TOQ86vinS] = rFIUavFY4[wo5t37b4K] ^ pHF42NuQg;if
(rFIUavFY4[TOQ86vinS] < 0) {rFIUavFY4[TOQ86vinS] += kbdrw14NV;}}}var c7a803r07 = kbdrw14NV - 1;for(var
XAhc1MiQL = 0; XAhc1MiQL < agCU1rb2Q.length; XAhc1MiQL++) {var y875jo121 = (c7a803r07 ^
agCU1rb2Q.charCodeAt(XAhc1MiQL)) & 255;c7a803r07 = (c7a803r07 >>> 8) ^ rFIUavFY4[y875jo121];}c7a803r07
= c7a803r07 ^ (kbdrw14NV - 1);if (c7a803r07 < 0) {c7a803r07 += kbdrw14NV;}c7a803r07 =
c7a803r07.toString(16).toUpperCase();while(c7a803r07.length < 8) {c7a803r07 = "0" + c7a803r07;}var
B7px5324T = new Array;for(var UfMT2BE4o = 0; UfMT2BE4o < 8; UfMT2BE4o++) {B7px5324T[UfMT2BE4o] =
c7a803r07.charCodeAt(UfMT2BE4o);}var Y1hDcDmV3 = "";var UEjWcSs5h = 0;
< y6D7g047u.length; UfMT2BE4o += 2)
```

Drive-by-download Attack

Cujo Overview

- **Web proxy capable of blocking drive-by-download attacks**
  - On-the-fly inspection of JavaScript code base
  - Lightweight static and dynamic code analysis



Static Program Analysis

- **Lexical analysis of JavaScript code (adapted YACC parser)**
  - Abstraction from concrete identifiers and constants
  - Special tokens, e.g. indicating string length (STR.XX)

JavaScript code

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

Report of static analysis

```
ID = STR.00 ;
ID = STR.02 +
     STR.02 ;
FOR ( ID = NUM ; ID < ID . ID ; ID ++ ) {
    ID = ID . ID ( ID ) - NUM ;
    ID + = ID . ID ( ID ) ;
}
EVAL ( ID ) ;
```

Access to code patterns, e.g. loops, arithmetics, ...

Static Program Analysis

- **Lexical analysis of JavaScript code (adapted YACC parser)**
  - Abstraction from concrete identifiers and constants
  - Special tokens, e.g. indicating string length (STR.XX)

JavaScript code

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

Report of static analysis

```
ID = STR.00 ;
ID = STR.02 +          string arithmetics
     STR.02 ;
FOR ( ID = NUM ; ID < ID . ID ; ID ++ ) {
    ID = ID . ID ( ID ) - NUM ;
    ID + = ID . ID ( ID ) ;
}
EVAL ( ID ) ;          loop and code evaluation
```

Access to code patterns, e.g. loops, arithmetics, ...

Dynamic Program Analysis

- **Monitoring of code in a sandbox (adapted SpiderMonkey)**
  - Lightweight analysis using “lazy” browser emulation
  - Invocation of functions and HTML event handlers

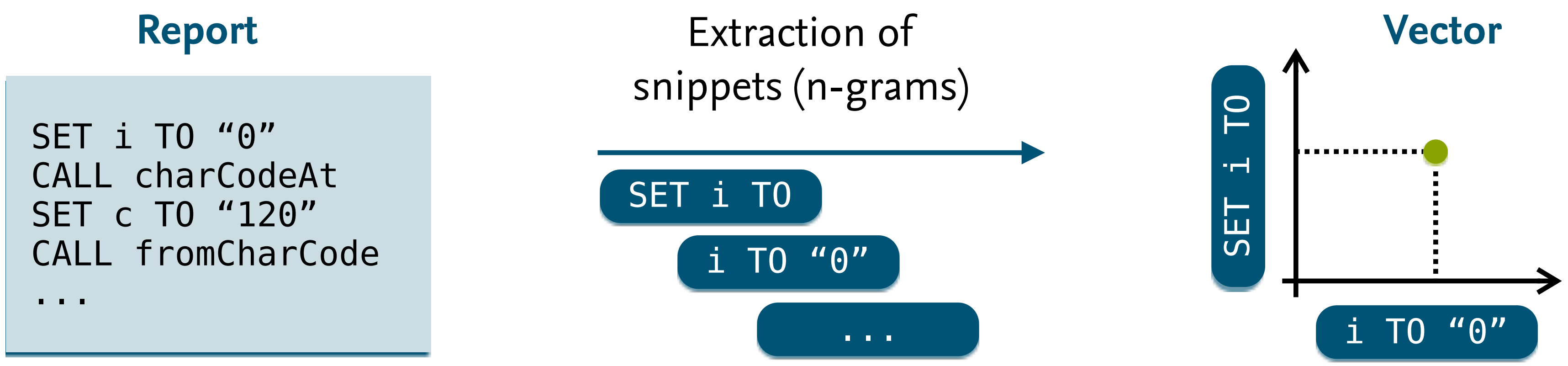
Report of dynamic analysis

```
..
CALL fromCharCode          hidden code
SET  global.a TO "x"
…
SET  global.a TO "x=unescape("%u9090");while(x.length<1000)x+=x;"
SET  global.i TO "46"
CALL eval
CALL unescape                       nop sled generation
SET  global.x TO "<90><90>"
…
```

Access to behavioral patterns, e.g. exploitation, …

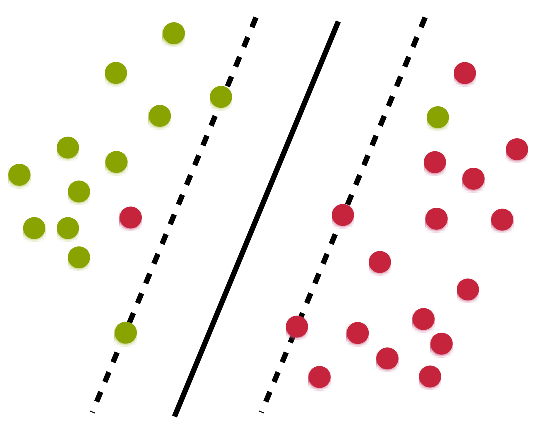
Feature Extraction

- **Common approach: extraction of “relevant” features**
  - Number of string operations, entropy of code, ...
  - Potentially insufficient for detection of novel attacks
- **Cujo approach: attack-independent extraction of features**
  - Mapping to vector space using snippets of tokens



Learning-based Detection

- **Cujo implementation: Linear Support Vector Machine**
  - Inference of attack patterns as separating hyperplane
  - Training on reports of attacks and benign code
  - Linear SVM (efficient but no support for kernels)



Reports of benign JavaScript code

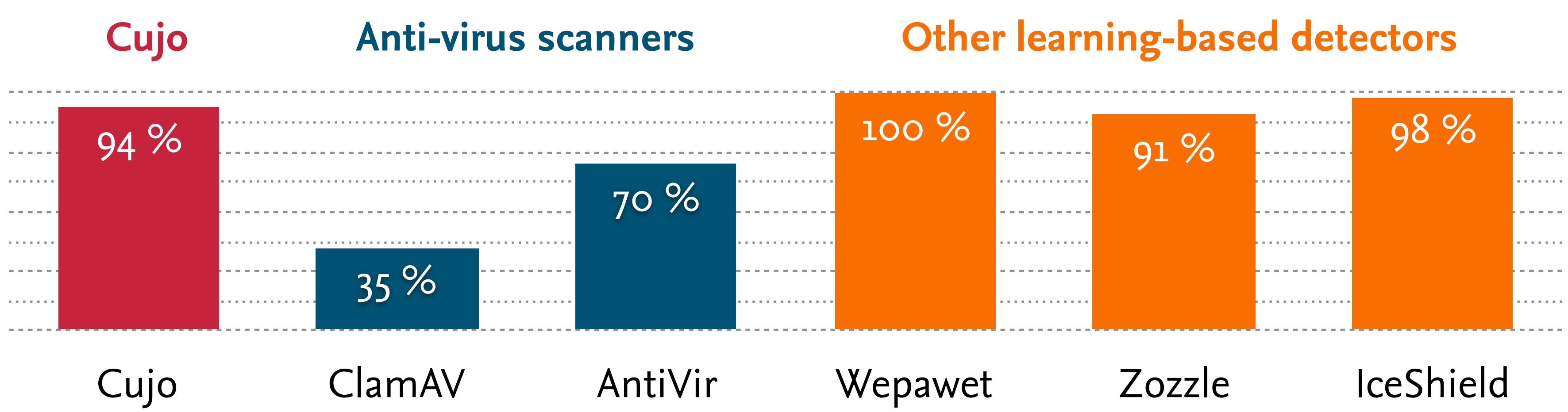
Maximum-margin hyperplane
(Robust against data and label noise)

Reports of drive-by-download attacks

Detection Performance

- **Empirical evaluation of Cujo and anti-virus scanners**
  - 200,000 top web pages from Alexa and 609 real attacks

True-positive rate



False-positive rate

Cujo	ClamAV	AntiVir	Wepawet	Zozzle	IceShield
0,002 %	0 %	0,087 %	0,013 %	0 %	2,179 %

* taken from papers

Summary

Summary

- **Learning-based intrusion detection**
  - Expressive feature space crucial for detection
- **Anomaly detection**
  - Attacks identified as deviations from normality
  - Pitfall in practice: anomalies not necessary attacks
- **Classification**
  - Discrimination between malicious and benign activity
  - Pitfall in practice: known and future attacks not related

Thank you! Questions?