Neue Herausforderungen in der Netzsicherheit

Tübingen 24.9.2015

Steffen Ullrich, genua

- Steffen Ullrich
- seit 2001 bei genua gmbh
- Entwicklung Highly Resistant Firewall genugate
- Forschungsprojekte zu Web 2.0 und gezielten Angriffen
  Padiofire, APT-Sweeper
  Unis Cottbus, Göttingen, Erlangen

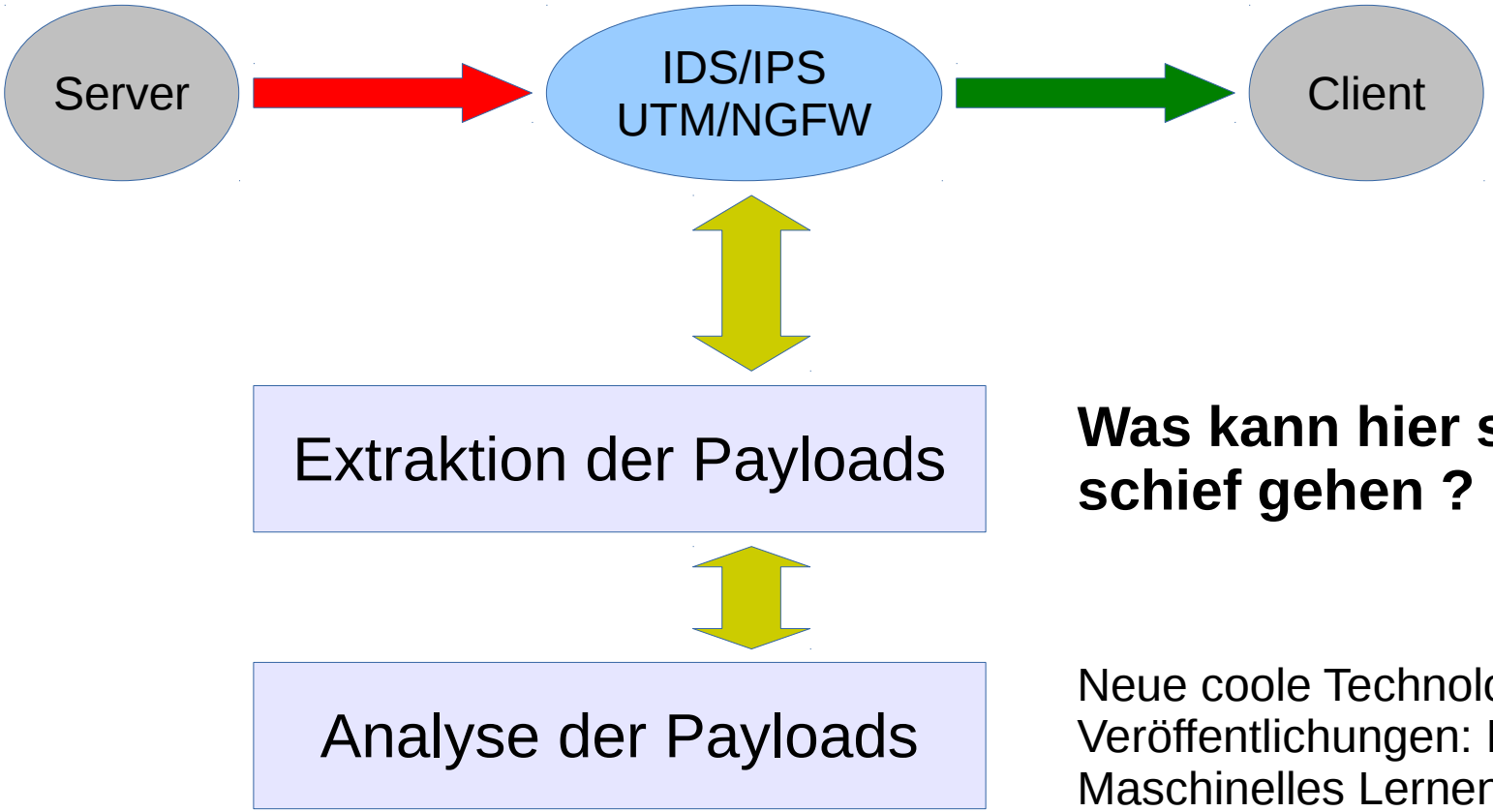
GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

genua

Alte und ungelöste ~~Neue~~ Herausforderungen in der Netzsicherheit

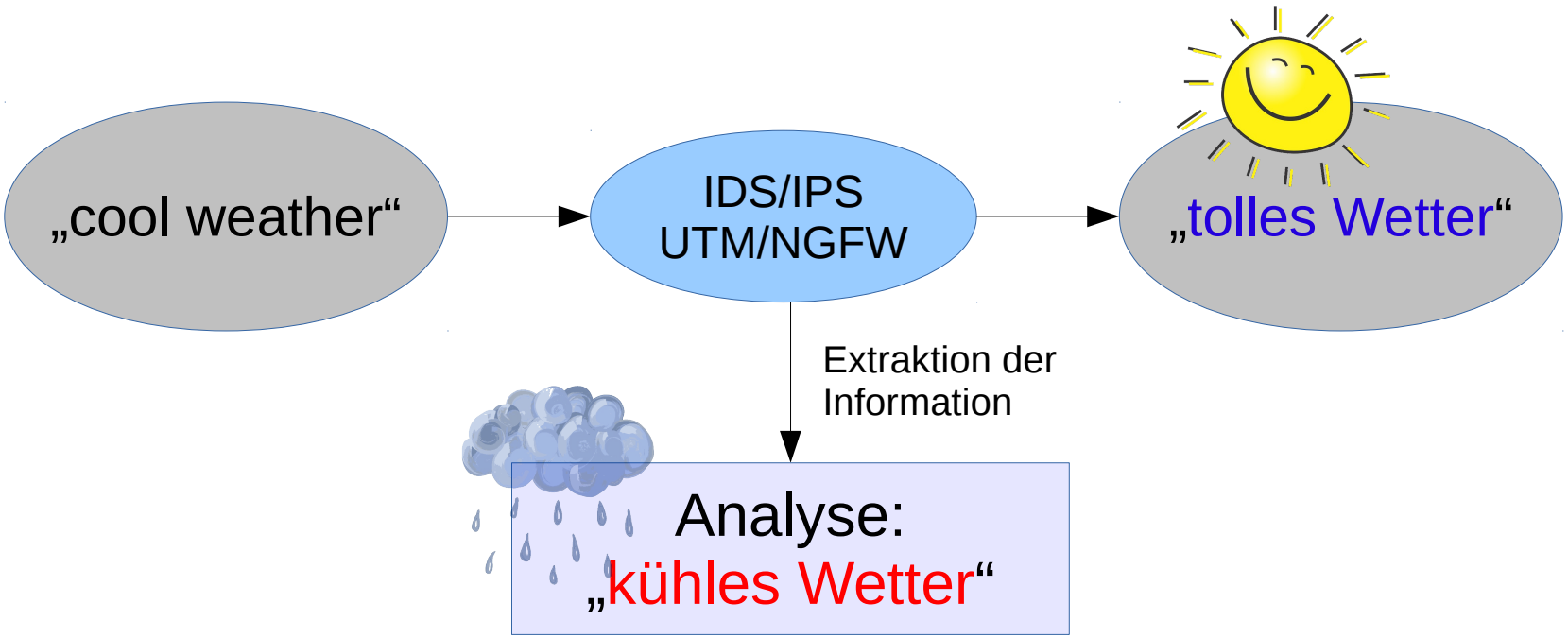
Umgehung von Sicherheitsanalysen durch Interpretationsdifferenzen

Erfassung, Extraktion, Analyse

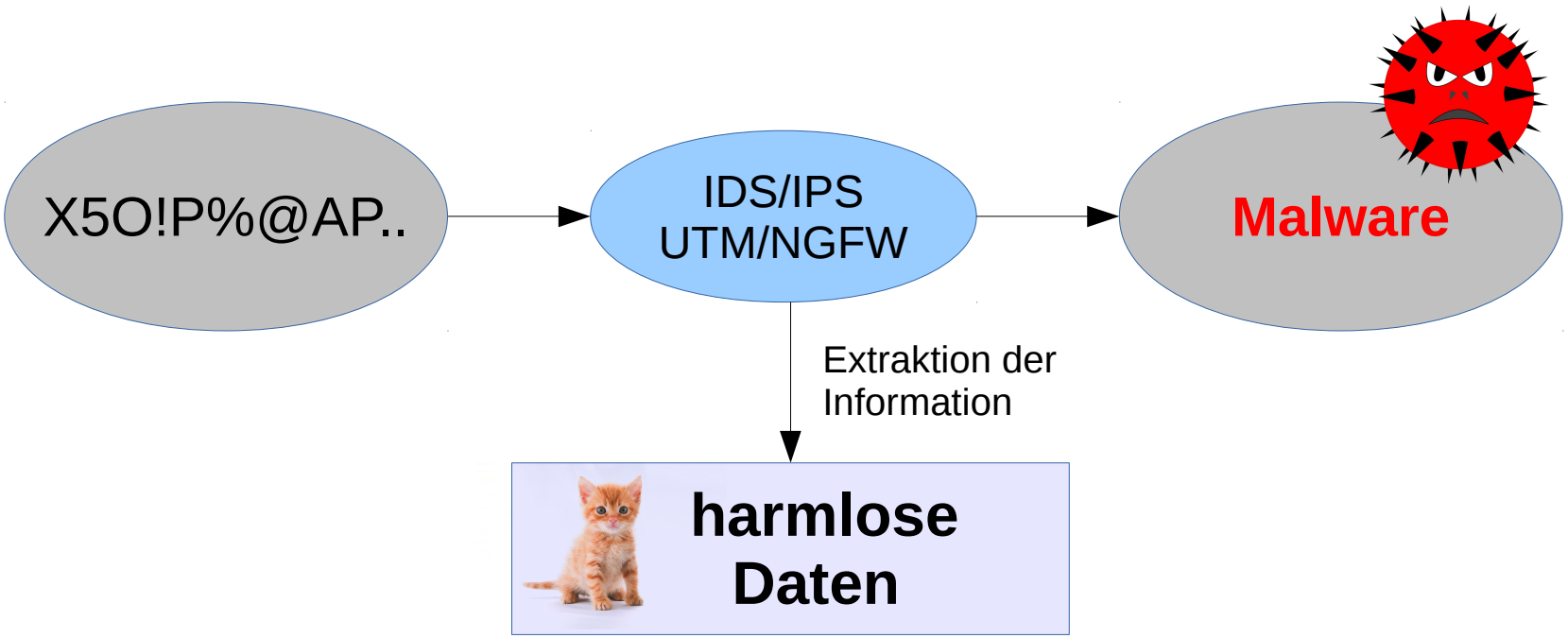


Was kann hier schon schief gehen ?

Neue coole Technologien und Veröffentlichungen: Big Data, Maschinelles Lernen, Buzzword#1, Buzzword#2, 99% Erkennungsrate ...



Interpretation des Datenstroms abhängig von Implementation und Kontext



Zuverlässiger Schutz benötigt gleiche Interpretation der Daten in Firewall und Client.

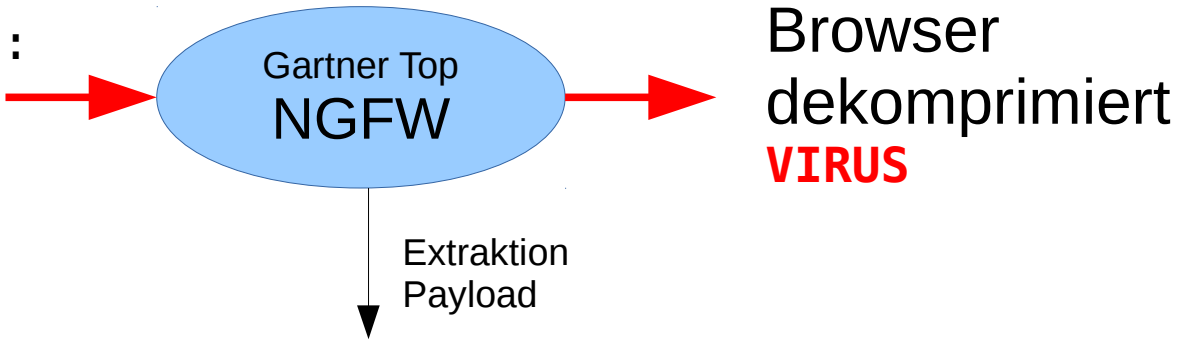
Semantic Gap durch Differenzen in der Implementation

Abweichende Interpretation* im Firewall führt zum Durchlassen des Virus.

* keine Unterstützung für deflate-Kompromierung, welche von allen Browsern unterstützt wird

```
HTTP/1.1 200 ok
Content-Encoding:
  deflate

deflate(VIRUS)
```



Firewall versteht kein deflate
analysiert Müll statt Virus

genua

- Testsuite für Umgehungen mit ungewöhnlichem oder kaputtem HTTP
- 2 Wochen laufen lassen
- mehr als 150 Testergebnisse
  mehr als 15 verschiedene Firewalls
- praktisch alle Produkte betroffen

genua

- 50% unterstützen keine Deflate-Kompression
  aber alle Browser unterstützen Deflate
- 40% wissen nicht, das Chunked-Encoding nur mit HTTP/1.1 geht
  alle Browser außer Safari verhalten sich richtig
- 40% versagen bei mehrfachen Content-Encoding Headern
  alle Browser außer IE verstehen es korrekt
- 35% analysieren keine HTTP 0.9 Antworten
  aber alle Browser verstehen diese
- **im Zweifelsfalle Daten durchlassen**

genua

- Browser verhalten sich anders als IPS
  → vielfältige Umgehungen möglich
- Alle getesten Produkte können trivial umgangen werden, meist mit vollständig validem HTTP
  Mindestens 6 Gartner Top NGFW dabei
- Selber testen → HTTP Evader
- IDS wie Snort, Bro, Suricata ebenso betroffen

Interpretationsdifferenzen MIME

```
Content-Transfer-Encoding: base64
Content-Transfer-Encoding: quoted-printable
Content-type: application/octet-stream;
    name='eicar.com'

WDV
PIVAlQEFQWzRcUFpYNTQoUF4pN0NDKTd9JEVJQ0FSLVNUQU5EQVJEL
U FOVElWSVJVUy1URVNU
LUZJTEUhJEgrSC
o=
```

AOL-Mail Virusscanner:

quoted-printable → Müll
Heuristik für base64 schlägt fehl wegen Leerzeichen, Zeilenlängen...

AOL-Mail Webinterface:

base64 → EICAR Test-Virus
wird heruntergeladen

genua

- Mail Clients interpretieren kaputtes MIME sehr unterschiedlich
- Konflikte zwischen Virenscanner und Webinterface bei Yahoo, GMX, AOL gefunden
- IDS Snort, Suricata, Bro sind blind
  Virenscanner durchwachsene Ergebnisse
  NGFW/UTM bisher nicht getestet

  → vielfältige Umgehungen möglich

Semantic Gap durch unvollständige Informationen zum Kontext

genua

ZIP ignoriert Junk am Anfang, Gzip am Ende

```
$ zip eicar.zip eicar.com
$ echo test | gzip > test.gz
$ cat test.gz eicar.zip > polyglot.zip

$ gzip -dc polyglot.zip
test
gzip: polyglot.zip: decompression OK, trailing garbage ignored

$ unzip -t polyglot.zip
warning [polyglot.zip]:  25 extra bytes at beginning or within zipfile
  (attempting to process anyway)
    testing: eicar.com                   OK
```

63% der Virenscanner auf virustotal versagen,
u.a. Fortinet, Kaspersky, Avira, ESET, Symantec ...

Abusing File Processing in Malware Detectors for Fun and Profit
https://www.cs.utexas.edu/~shmat/shmat_oak12av.pdf

genua

Lokale Endung entscheidet i.A. über Behandlung im Betriebssystem. Download mit passendem Namen kann erzwungen werden.

```
<!doctype html>
<a href=eicar.gif download=eicar.zip>
```

Endung von Link und gelieferter Content-Type egal. IPS weiß nicht was Client mit Inhalt anfängt.
Accept-Header in Request liefert in einigen Fällen eine Idee was der Client will.

```
HTTP/1.1 200 ok
Content-type: image/gif

{junk.gif}{eicar.zip}
```

genua

- Mehrfache und widersprüchliche Angaben zur Zeichenkodierung möglich
- IPS sieht nur ein Teil dieses Kontexts
  - HTTP-Header
    `Content-type: text/javascript; charset=UTF-7`
  - Byte-Order-Mark U+FEFF
- und ist blind für den Rest
  - charset-Attribute beim Einbetten
    `<script charset=UTF-7 src=foo.js>`
  - nichts angegeben: vererbt von einbettender Seite, Browserdefault, raten

genua

- Script-Tag forciert Kontext aber sagt dieses dem Server nicht. Wert des gelieferten Content-type wird ignoriert.

```
<!doctype html>
<script charset=utf-7 src=script.txt>
```

- IPS sieht nur Text, weiß nicht das UTF-7 kodiert (spezifisch MSIE).

```
HTTP/1.0 200 ok
Content-type: text/plain

+AGEAbABlAHIAdAAA-('+AHAATwB3AG4AZQBk-!');
```

alert('pOwned!');

genua

Viele (insb. mobile) Mail-Clients holen bei IMAP zuerst die MIME-Struktur:

```
26 UID FETCH 12 (UID BODYSTRUCTURE)
* 1 FETCH (UID 12 BODYSTRUCTURE ...
   ("application" "octet-stream"
   ("name" "eicar.com") NIL NIL "base64" 100 ....
```

Später (andere Verbindung) wird der Body geholt. IDS weiß nicht, das dieser mit base64 kodiert ist.

```
1 FETCH (UID 14 BODY[3] {100}
WDV
PIVAlQEFQWzRcUFpYNTQoUF4pN0NDKTd9JEVJQ0FSLV
NUQU 5EQVJELUFOVElWSVJVUy1URVNU
LUZJTEUhJEgrSC
o=)
26 OK Fetch completed.
```

genua

Inhalte werden lokal gecached und später in anderem Kontext erneut benutzt.

```
-- img-script.html
<!doctype html>
<img src=eicar.gif>
<a href=img-script2.html>
     click</a>
```

Lade eicar.gif vom Server *
benutze als Bild

* IPS sieht evtl. Accept: image/..

Lade img-script2.html vom Server

```
-- img-script2.html
<!doctype html>
<script src=eicar.gif>
```

benutze eicar.gif aus Cache*
aber benutze als Script

* IPS sieht nichts

Kontextabhängige Codeausführung zur Umgehung dynamischer Analysen

load.html

```
<script src=script.php>
```

fire.html

```
<script src=script.php>
<script>attack()</script>
```

script.php – Referer load.html

```
function attack() {
... irgendwas Böses ...
}
```

script.php – Referer fire.html

```
function attack() {
... süßes Kätzchen...
}
```

load
lädt **script.php** von Server

fire
nutzt **script.php** aus Cache

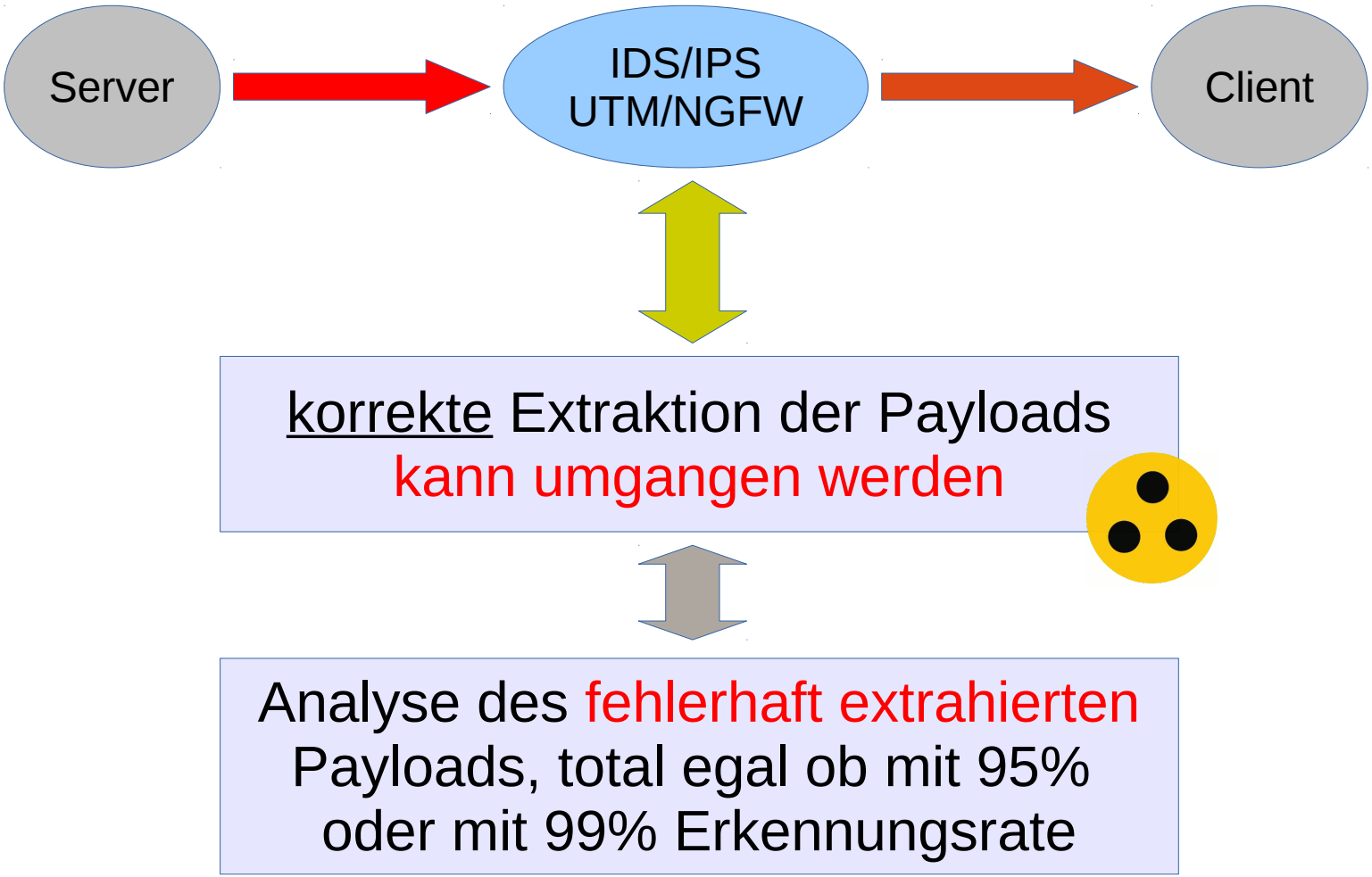
fire
lädt **script.php** von Server

genua

- Codeausführung abhängig
  - von gecachten Inhalten
  - von spezifischen Cookies
  - ob eingeloggt bei Facebook, Google, Twitter, Intranet-Wiki...
  - Nutzerreaktionen (Scrollen, Klicks...)
  - ...

- Semantic Gap ist trivial ausnutzbar
- kann bei Implementationsdifferenzen durch aktive Bereinigung der Daten gefixt werden (d.h. aktiver Proxy, kein passives IDS)
- bei unbekanntem Kontext per Design kaum Fix möglich
- Herausforderung ist es
  - Umgehungsversuche sicher zu erkennen
  - Trotz Semantic Gap zuverlässige Analysen zu ermöglichen

Zusammenfassung



Fragen?