Techniken zur Identifizierung von Netzwerk-Protokollen

Florian Adamsky

27C3: We come in Peace

28. Dezember 2010

Inhaltsverzeichnis

Einleitung

- suchte spannendes Thema für meine Bachelorthesis
- Zugangserschwerungsgesetz
- intelligente Dienstgüte (QoS) für Heimnetzwerke

Einleitung

- suchte spannendes Thema für meine Bachelorthesis
- Zugangserschwerungsgesetz
- intelligente Dienstgüte (QoS) für Heimnetzwerke

Einleitung

- suchte spannendes Thema für meine Bachelorthesis
- Zugangserschwerungsgesetz
- intelligente Dienstgüte (QoS) für Heimnetzwerke

Prämisse

QoSiLAN = Quality of Service for Local Area Networks

1. Phase
Analysieren des Netzwerks

2. Phase
Identifizieren der Protokolle

3. Phase
Reservieren und Priorisieren der Bandbreite

Anforderungen

- Nahe Echtzeit Erkennung
- Fokus auf Streaming-Protokolle
- Portabilität

Prämisse

QoSiLAN = Quality of Service for Local Area Networks

1. Phase
Analysieren des Netzwerks

2. Phase
Identifizieren der Protokolle

3. Phase
Reservieren und Priorisieren der Bandbreite

Anforderungen

- Nahe Echtzeit Erkennung
- Fokus auf Streaming-Protokolle
- Portabilität

Prämisse

QoSiLAN = Quality of Service for Local Area Networks

1. Phase
Analysieren des Netzwerks

2. Phase
Identifizieren der Protokolle

3. Phase
Reservieren und Priorisieren der Bandbreite

Anforderungen

- Nahe Echtzeit Erkennung
- Fokus auf Streaming-Protokolle
- Portabilität

Prämisse

QoSiLAN = Quality of Service for Local Area Networks

1. Phase	2. Phase	3. Phase
Analysieren des Netzwerks	Identifizieren der Protokolle	Reservieren und Priorisieren der Bandbreite

Anforderungen

- Niedrige Echtzeit Erkennung
- Fokus auf Streaming-Protokolle
- Portabilität

Prämisse

QoSiLAN = Quality of Service for Local Area Networks

1. Phase	2. Phase	3. Phase
Analysieren des Netzwerks	Identifizieren der Protokolle	Reservieren und Priorisieren der Bandbreite

Anforderungen

- Nahe-Echtzeit Erkennung
- Fokus auf Streaming-Protokolle
- Portabilität

Prämisse

QoSiLAN = Quality of Service for Local Area Networks

1. Phase	2. Phase	3. Phase
Analysieren des Netzwerks	Identifizieren der Protokolle	Reservieren und Priorisieren der Bandbreite

Anforderungen

- Nahe-Echtzeit Erkennung
- Fokus auf Streaming-Protokolle
- Portabilität

Prämisse

QoSiLAN = Quality of Service for Local Area Networks

1. Phase	2. Phase	3. Phase
Analysieren des Netzwerks	Identifizieren der Protokolle	Reservieren und Priorisieren der Bandbreite

Anforderungen

- Nahe-Echtzeit Erkennung
- Fokus auf Streaming-Protokolle
- Portabilität

Inhaltsverzeichnis

Netzwerkprotokoll Identifizierung

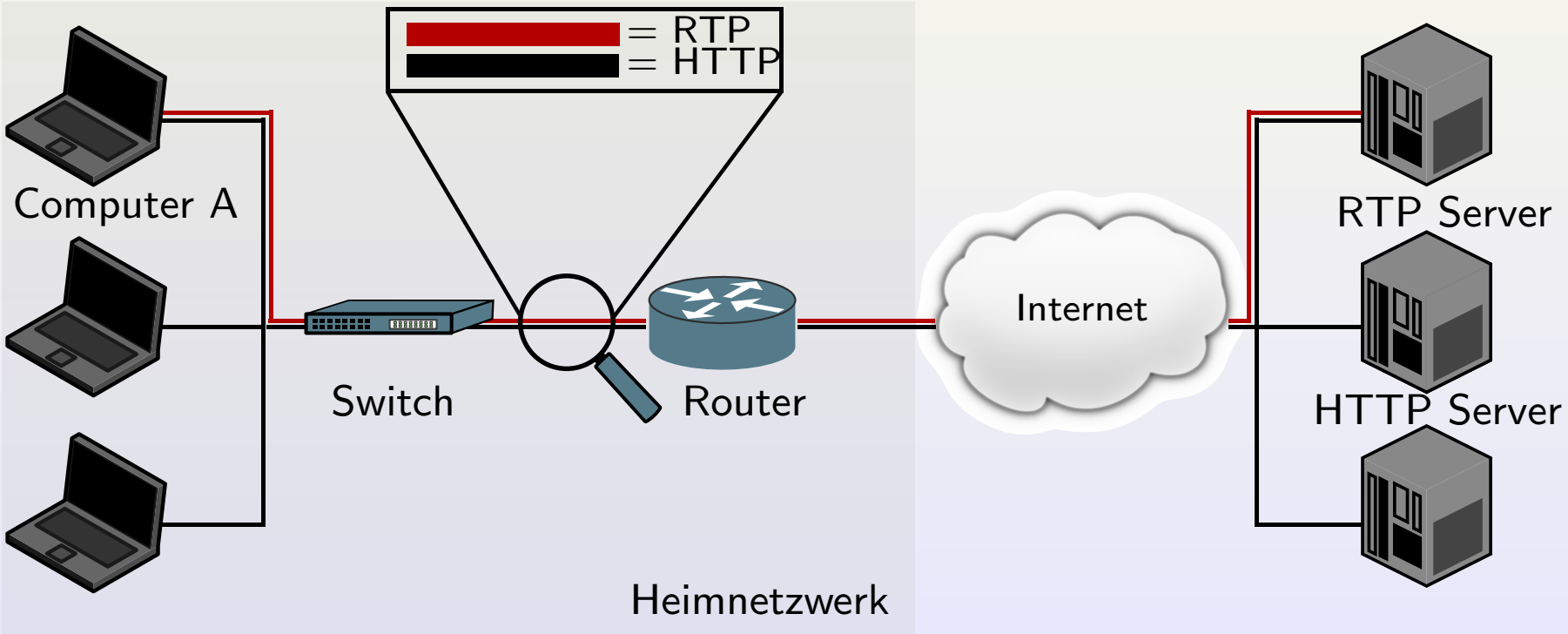
Definition

Identifizierung des Protokolls, welches auf Layer 7 des OSI-Modells oder auf Layer 4 des TCP/IP-Modells läuft.

Netzwerkprotokoll Identifizierung

Definition

Identifizierung des Protokolls, welches auf Layer 7 des OSI-Modells oder auf Layer 4 des TCP/IP-Modells läuft.



Weitere Begriffe

- Application Identification
- Port Independent Protocol Identification (PIPI)
- Protocol Discovery
- Application Recognition
- Traffic Classification

Weitere Begriffe

- Application Identification
- Port Independent Protocol Identification (PIPI)
- Protocol Discovery
- Application Recognition
- Traffic Classification

Weitere Begriffe

- Application Identification
- Port Independent Protocol Identification (PIPI)
- Protocol Discovery
- Application Recognition
- Traffic Classification

Weitere Begriffe

- Application Identification
- Port Independent Protocol Identification (PIPI)
- Protocol Discovery
- Application Recognition
- Traffic Classification

Weitere Begriffe

- Application Identification
- Port Independent Protocol Identification (PIPI)
- Protocol Discovery
- Application Recognition
- Traffic Classification

Anforderungen an die Techniken

- Hohe Trefferquote mit wenigen oder keinen Falsch-Positiven
- Echtzeit-Erkennung unter hoher Netzwerklast
- Robust unter verschiedenen Netzwerkkonfigurationen
- Einfache Möglichkeit neue Protokolle zu lernen
- Resistent gegen Protokoll-Obfuskation

Anforderungen an die Techniken

- Hohe Trefferquote mit wenigen oder keinen Falsch-Positiven
- Echtzeit-Erkennung unter hoher Netzwerklast
- Robust unter verschiedenen Netzwerkkonfigurationen
- Einfache Möglichkeit neue Protokolle zu lernen
- Resistent gegen Protokoll-Obfuskation

Anforderungen an die Techniken

- Hohe Trefferquote mit wenigen oder keinen Falsch-Positiven
- Echtzeit-Erkennung unter hoher Netzwerklast
- Robust unter verschiedenen Netzwerkkonfigurationen
- Einfache Möglichkeit neue Protokolle zu lernen
- Resistent gegen Protokoll-Obfuskation

Anforderungen an die Techniken

- Hohe Trefferquote mit wenigen oder keinen Falsch-Positiven
- Echtzeit-Erkennung unter hoher Netzwerklast
- Robust unter verschiedenen Netzwerkkonfigurationen
- Einfache Möglichkeit neue Protokolle zu lernen
- Resistent gegen Protokoll-Obfuskation

Anforderungen an die Techniken

- Hohe Trefferquote mit wenigen oder keinen Falsch-Positiven
- Echtzeit-Erkennung unter hoher Netzwerklast
- Robust unter verschiedenen Netzwerkkonfigurationen
- Einfache Möglichkeit neue Protokolle zu lernen
- Resistent gegen Protokoll-Obfuskation

Einsatzmöglichkeit

Netzwerk

- Bandbreitenmanagement mit QoS
- Security durch Intrusion Detection Systems (IDS) oder Intrusion Prevention Systems (IPS)
- graduelle Datentarife und Ad-Injection

politische und soziale Kontrolle

- abhören und überwachen des Internetverkehrs
- filtern und zensieren von unliebsamen Inhalten

Einsatzmöglichkeit

Netzwerk

- Bandbreitenmanagement mit QoS
- Security durch Intrusion Detection Systems (IDS) oder Intrusion Prevention Systems (IPS)
- graduelle Datentarife und Ad-Injection

politische und soziale Kontrolle

- abhören und überwachen des Internetverkehrs
- filtern und zensieren von unliebsamen Inhalten

Einsatzmöglichkeit

Netzwerk

- Bandbreitenmanagement mit QoS
- Security durch Intrusion Detection Systems (IDS) oder Intrusion Prevention Systems (IPS)
- graduelle Datentarife und Ad-Injection

politische und soziale Kontrolle

- abhören und überwachen des Internetverkehrs
- filtern und zensieren von unliebsamen Inhalten

Einsatzmöglichkeit

Netzwerk

- Bandbreitenmanagement mit QoS
- Security durch Intrusion Detection Systems (IDS) oder Intrusion Prevention Systems (IPS)
- graduelle Datentarife und Ad-Injection

politische und soziale Kontrolle

- abhören und überwachen des Internetverkehrs
- filtern und zensieren von unliebsamen Inhalten

Einsatzmöglichkeit

Netzwerk

- Bandbreitenmanagement mit QoS
- Security durch Intrusion Detection Systems (IDS) oder Intrusion Prevention Systems (IPS)
- graduelle Datentarife und Ad-Injection

politische und soziale Kontrolle

- abhören und überwachen des Internetverkehrs
- filtern und zensieren von unliebsamen Inhalten

Techniken

1. TCP/UDP Portnummern
2. Deep Packet Inspection
3. Maschinen-Lern-Algorithmen
4. Hybrid-Formen

TCP/UDP Portnummern

Theorem

Portnummern sind keine robuste Methode zur Identifizierung.

Beweis.

- Studie: rund 50–70 % des Internet-Verkehrs kann identifiziert werden
- viele Anwendungen haben keine eindeutigen Portnummern (z.B. P2P)
- einige Ports sind mehrfach vergeben (z.B. TCP Port 888)
- Firewalls und NAT sind schuld, dass fast alles über Port 80 kommuniziert
- Angriffe sind üblicherweise nicht mit einem Port verknüpft

□

TCP/UDP Portnummern

Theorem

Portnummern sind keine robuste Methode zur Identifizierung.

Beweis.

- Studie: rund 50–70 % des Internet-Verkehrs kann identifiziert werden
- viele Anwendungen haben keine eindeutigen Portnummern (z.B. P2P)
- einige Ports sind mehrfach vergeben (z.B. TCP Port 888)
- Firewalls und NAT sind schuld, dass fast alles über Port 80 kommuniziert
- Angriffe sind üblicherweise nicht mit einem Port verknüpft

□

TCP/UDP Portnummern

Theorem

Portnummern sind keine robuste Methode zur Identifizierung.

Beweis.

- Studie: rund 50–70 % des Internet-Verkehrs kann identifiziert werden
- viele Anwendungen haben keine eindeutigen Portnummern (z.B. P2P)
- einige Ports sind mehrfach vergeben (z.B. TCP Port 888)
- Firewalls und NAT sind schuld, dass fast alles über Port 80 kommuniziert
- Angriffe sind üblicherweise nicht mit einem Port verknüpft

□

TCP/UDP Portnummern

Theorem

Portnummern sind keine robuste Methode zur Identifizierung.

Beweis.

- Studie: rund 50–70 % des Internet-Verkehrs kann identifiziert werden
- viele Anwendungen haben keine eindeutigen Portnummern (z.B. P2P)
- einige Ports sind mehrfach vergeben (z.B. TCP Port 888)
- Firewalls und NAT sind schuld, dass fast alles über Port 80 kommuniziert
- Angriffe sind üblicherweise nicht mit einem Port verknüpft

□

TCP/UDP Portnummern

Theorem

Portnummern sind keine robuste Methode zur Identifizierung.

Beweis.

- Studie: rund 50–70 % des Internet-Verkehrs kann identifiziert werden
- viele Anwendungen haben keine eindeutigen Portnummern (z.B. P2P)
- einige Ports sind mehrfach vergeben (z.B. TCP Port 888)
- Firewalls und NAT sind schuld, dass fast alles über Port 80 kommuniziert
- Angriffe sind üblicherweise nicht mit einem Port verknüpft

□

TCP/UDP Portnummern

Theorem

Portnummern sind keine robuste Methode zur Identifizierung.

Beweis.

- Studie: rund 50–70 % des Internet-Verkehrs kann identifiziert werden
- viele Anwendungen haben keine eindeutigen Portnummern (z.B. P2P)
- einige Ports sind mehrfach vergeben (z.B. TCP Port 888)
- Firewalls und NAT sind schuld, dass fast alles über Port 80 kommuniziert
- Angriffe sind üblicherweise nicht mit einem Port verknüpft

□

Deep Packet Inspection

Definition

Die DPI identifziert Protokolle mit *multi-pattern matching* Algorithmen. Bedeutet nicht "*tief ins Paket*" schauen.

- Pattern wird "*application signature*" genannt
- Im Moment die zuverlässigste Methode um Protokolle zu erkennen
- Hohe Verbreitung in kommerziellen Produkten

Deep Packet Inspection

Definition

Die DPI identifziert Protokolle mit *multi-pattern matching* Algorithmen. Bedeutet nicht "*tief ins Paket*" schauen.

- Pattern wird "*application signature*" genannt
- Im Moment die zuverlässigste Methode um Protokolle zu erkennen
- hohe Verbreitung in kommerziellen Produkten

Deep Packet Inspection

Definition

Die DPI identifziert Protokolle mit *multi-pattern matching* Algorithmen. Bedeutet nicht "*tief ins Paket*" schauen.

- Pattern wird "*application signature*" genannt
- Im Moment die zuverlässigste Methode um Protokolle zu erkennen
- hohe Verbreitung in kommerziellen Produkten

Deep Packet Inspection

Definition

Die DPI identifziert Protokolle mit *multi-pattern matching* Algorithmen. Bedeutet nicht "*tief ins Paket*" schauen.

- Pattern wird "*application signature*" genannt
- Im Moment die zuverlässigste Methode um Protokolle zu erkennen
- hohe Verbreitung in kommerziellen Produkten

Deep Packet Inspection

String matching

exakter Treffer

- Aho-Corasick
- Wu-Manber
- SBOM

annähernde Treffer

- Bloomfilter

reguläre Ausdrücke

- L7-Filter

Deep Packet Inspection

String matching

exakter Treffer

- Aho-Corasick
- Wu-Manber
- SBOM

annähernde Treffer

- Bloomfilter

reguläre Ausdrücke

- L7-Filter

Deep Packet Inspection

String matching

exakter Treffer

- Aho-Corasick
- Wu-Manber
- SBOM

annähernde Treffer

- Bloomfilter

reguläre Ausdrücke

- L7-Filter

Deep Packet Inspection

Probleme

- keine Identifizierung, wenn Payload nicht lesbar ist
- hoher Aufwand notwendig, um neue Protokolle hinzuzufügen
  - Dokumentation studieren und Gesetzmäßigkeiten finden
  - Gesetzmäßigkeiten müssen gepflegt werden
- Es gibt Ansätze um automatisiert nach “Application Signature” zu suchen
- hohes Missbrauchspotenzial

Deep Packet Inspection

Probleme

- keine Identifizierung, wenn Payload nicht lesbar ist
- hoher Aufwand notwendig, um neue Protokolle hinzuzufügen
  - Dokumentation studieren und Gesetzmäßigkeiten finden
  - Gesetzmäßigkeiten müssen gepflegt werden
- Es gibt Ansätze um automatisiert nach "Application Signature" zu suchen
- hohes Missbrauchspotenzial

Deep Packet Inspection

Probleme

- keine Identifizierung, wenn Payload nicht lesbar ist
- hoher Aufwand notwendig, um neue Protokolle hinzuzufügen
  - Dokumentation studieren und Gesetzmäßigkeiten finden
  - Gesetzmäßigkeiten müssen gepflegt werden
- Es gibt Ansätze um automatisiert nach "Application Signature" zu suchen
- hohes Missbrauchspotenzial

Deep Packet Inspection

Probleme

- keine Identifizierung, wenn Payload nicht lesbar ist
- hoher Aufwand notwendig, um neue Protokolle hinzuzufügen
  - Dokumentation studieren und Gesetzmäßigkeiten finden
  - Gesetzmäßigkeiten müssen gepflegt werden
- Es gibt Ansätze um automatisiert nach “Application Signature” zu suchen
- hohes Missbrauchspotenzial

Deep Packet Inspection

Probleme

- keine Identifizierung, wenn Payload nicht lesbar ist
- hoher Aufwand notwendig, um neue Protokolle hinzuzufügen
  - Dokumentation studieren und Gesetzmäßigkeiten finden
  - Gesetzmäßigkeiten müssen gepflegt werden
- Es gibt Ansätze um automatisiert nach “Application Signature” zu suchen
- hohes Missbrauchspotenzial

Deep Packet Inspection

Probleme

- keine Identifizierung, wenn Payload nicht lesbar ist
- hoher Aufwand notwendig, um neue Protokolle hinzuzufügen
  - Dokumentation studieren und Gesetzmäßigkeiten finden
  - Gesetzmäßigkeiten müssen gepflegt werden
- Es gibt Ansätze um automatisiert nach “Application Signature” zu suchen
- hohes Missbrauchspotenzial

Maschinen-Lern-Algorithmen

Überwachtes Lernen

- Support Vector Machine
- Decision Tree
- Naive Bayes

Unüberwachtes Lernen

- Neuronale Netze

Probleme

- mathematisch komplex, daher aufwendig zu berechnen
- benötigen meist abgeschlossene Flows, sind daher unbrauchbar für Echtzeit-Identifizierung

Maschinen-Lern-Algorithmen

Überwachtes Lernen

- Support Vector Machine
- Decision Tree
- Naive Bayes

Unüberwachtes Lernen

- Neuronale Netze

Probleme

- mathematisch komplex, daher aufwendig zu berechnen
- benötigen meist abgeschlossene Flows, sind daher unbrauchbar für Echtzeit-Identifizierung

Maschinen-Lern-Algorithmen

Überwachtes Lernen

- Support Vector Machine
- Decision Tree
- Naive Bayes

Unüberwachtes Lernen

- Neuronale Netze

Probleme

- mathematisch komplex; daher aufwendig zu berechnen
- benötigen meist abgeschlossene Flows, sind daher unbrauchbar für Echtzeit-Identifizierung

Maschinen-Lern-Algorithmen

Überwachtes Lernen

- Support Vector Machine
- Decision Tree
- Naive Bayes

Unüberwachtes Lernen

- Neuronale Netze

Probleme

- mathematisch komplex; daher aufwendig zu berechnen
- benötigen meist abgeschlossene Flows, sind daher unbrauchbar für Echtzeit-Identifizierung

Hybrid-Formen

- werden vermehrt verwendet

Beispiel

- bekannte Protokolle mit DPI
- unbekannte Protokolle mit Statistik

Hybrid-Formen

- werden vermehrt verwendet

Beispiel

- bekannte Protokolle mit DPI
- unbekannte Protokolle mit Statistik

Hybrid-Formen

- werden vermehrt verwendet

Beispiel

- bekannte Protokolle mit DPI
- unbekannte Protokolle mit Statistik

Inhaltsverzeichnis

Wer hat's erfunden?

- SPID: Statistical Protocol IDentification
- 2009 entwickelt von Erik Hjelmvik und Wolfgang John

Wer hat's erfunden?

- SPID: Statistical Protocol IDentification
- 2009 entwickelt von Erik Hjelmvik und Wolfgang John

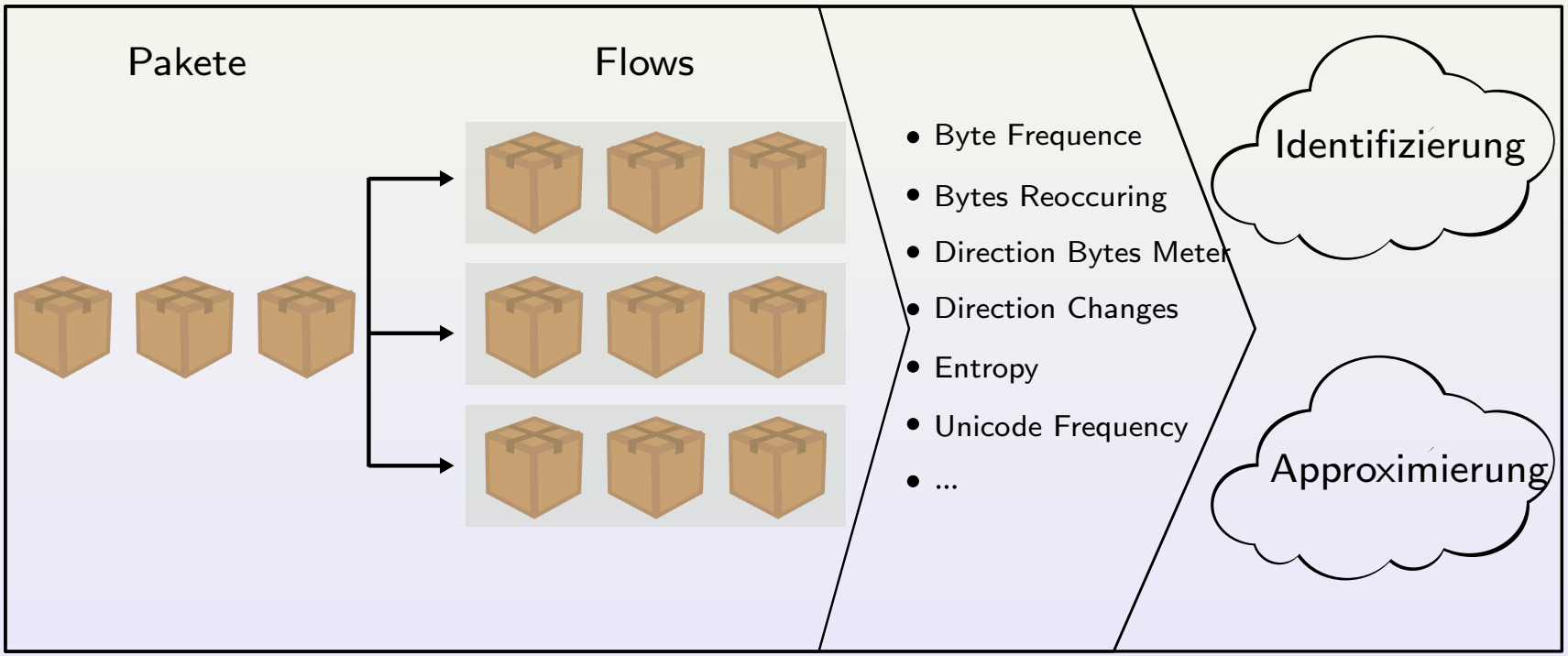
Wer hat's erfunden?

- SPID: Statistical Protocol IDentification
- 2009 entwickelt von Erik Hjelmvik und Wolfgang John

Wer hat's erfunden?

- SPID: Statistical Protocol IDentification
- 2009 entwickelt von Erik Hjelmvik und Wolfgang John

1. Schritt: Gruppierung 2. Schritt: Analyse 3. Schritt: Operation



Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Gruppierung

- Analyse auf bidirektionalen Flows
- Flows werden erkannt durch 5-Tupel
- 5-Tupel ist definiert:
  - source IP
  - source Port
  - destination IP
  - destination Port
  - transport protokol

TCP

- Flow beginnt erst nach Three-Way-Handshake
  - Kaltstart-Problem
- Pure ACKs werden ignoriert, da kein Payload enthalten ist

Identifizierung

Operation: Kullback-Leibler Divergenz

Die KL-Divergenz – auch relative Entropie – ist ein Maß für die Unterschiedlichkeit zweier Wahrscheinlichkeitsverteilungen.

$$D(P\|Q) = KL(P, Q) = \sum_{x \in X} P(x) \log_2 \frac{P(x)}{Q(x)}$$

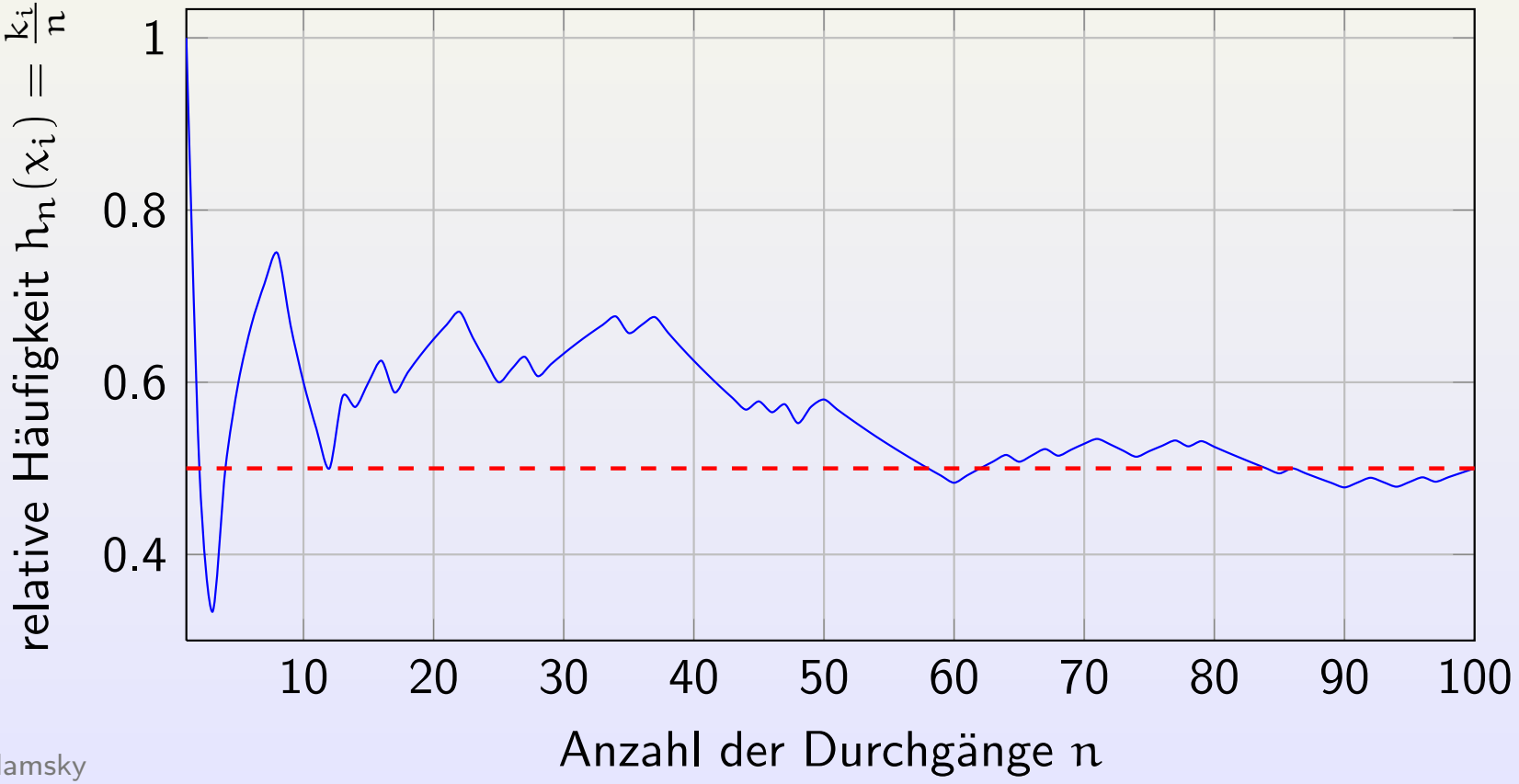
Eigenschaften

- P = trainierte Flows
- Q = vorbeifließende Flows
- nicht symmetrisch: $KL(P, Q) \neq KL(Q, P)$
- Wenn $P = Q$, dann gilt $D(P\|Q) = 0$, ansonsten $D(P\|Q) > 0$

Gesetz der großen Zahlen

Die relative Häufigkeit eines zufälligen Ereignisses weicht für immer längere Versuchsserien beliebig von einem festen Grenzwert ab.

$$h_n(E) \to P(E)$$



Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Implementierungen

- PoC in C# von Erik Hjelmvik mit über 30 Messverfahren (`http://sourceforge.net/apps/mediawiki/spid/`)
- Fork in C++ mit libPcap für Tests auf SoHo Geräten
- neue Leistungsmerkmale:
  - Unterstützung des UDP Transport-Protokolls
  - Echtzeit-Erkennung (live-capturing)
  - optimierte Datenbank
  - ausgewählte Messverfahren

Git Repository

```
git clone git@github.com:cit/Spid.git
```

Messverfahren

- Byte-Frequenz
- Anzahl der Richtungswechsel
- Datenmenge pro Richtung
- Hashfunktion über die ersten 4 Bytes
- Aktion und Reaktion
- Entropie
- Periodizität von Byte-Paaren
- Similarität der ersten drei Bytes
- Unicode-Frequenz
- Bit-Frequenz
- Payload-Größe des ersten Pakets

Messverfahren

- Byte-Frequenz
- Anzahl der Richtungswechsel
- Datenmenge pro Richtung
- Hashfunktion über die ersten 4 Bytes
- Aktion und Reaktion
- Entropie
- Periodizität von Byte-Paaren
- Similarität der ersten drei Bytes
- Unicode-Frequenz
- Bit-Frequenz
- Payload-Größe des ersten Pakets

Messverfahren: Byte-Frequenz

- auch Häufigkeitsanalyse genannt
- operiert auf dem ersten TCP-Paket in jede Richtung
- zählt die relative Häufigkeit eines Bytes in Bezug auf seine Größe

Beispiel

HTTP

```
47 45 54 20   2F 66 6F 6F   G E T   / f o o
48 54 54 50   2F 31 2E 31   H T T P / 1 . 1
```

Messverfahren: Byte-Frequenz

- auch Häufigkeitsanalyse genannt
- operiert auf dem ersten TCP-Paket in jede Richtung
- zählt die relative Häufigkeit eines Bytes in Bezug auf seine Größe

Beispiel

HTTP

```
47 45 54 20   2F 66 6F 6F   G E T   / f o o
48 54 54 50   2F 31 2E 31   H T T P / 1 . 1
```

Messverfahren: Byte-Frequenz

- auch Häufigkeitsanalyse genannt
- operiert auf dem ersten TCP-Paket in jede Richtung
- zählt die relative Häufigkeit eines Bytes in Bezug auf seine Größe

Beispiel

HTTP

```
47 45 54 20   2F 66 6F 6F   G E T   / f o o
48 54 54 50   2F 31 2E 31   H T T P / 1 . 1
```

Messverfahren: Byte-Frequenz

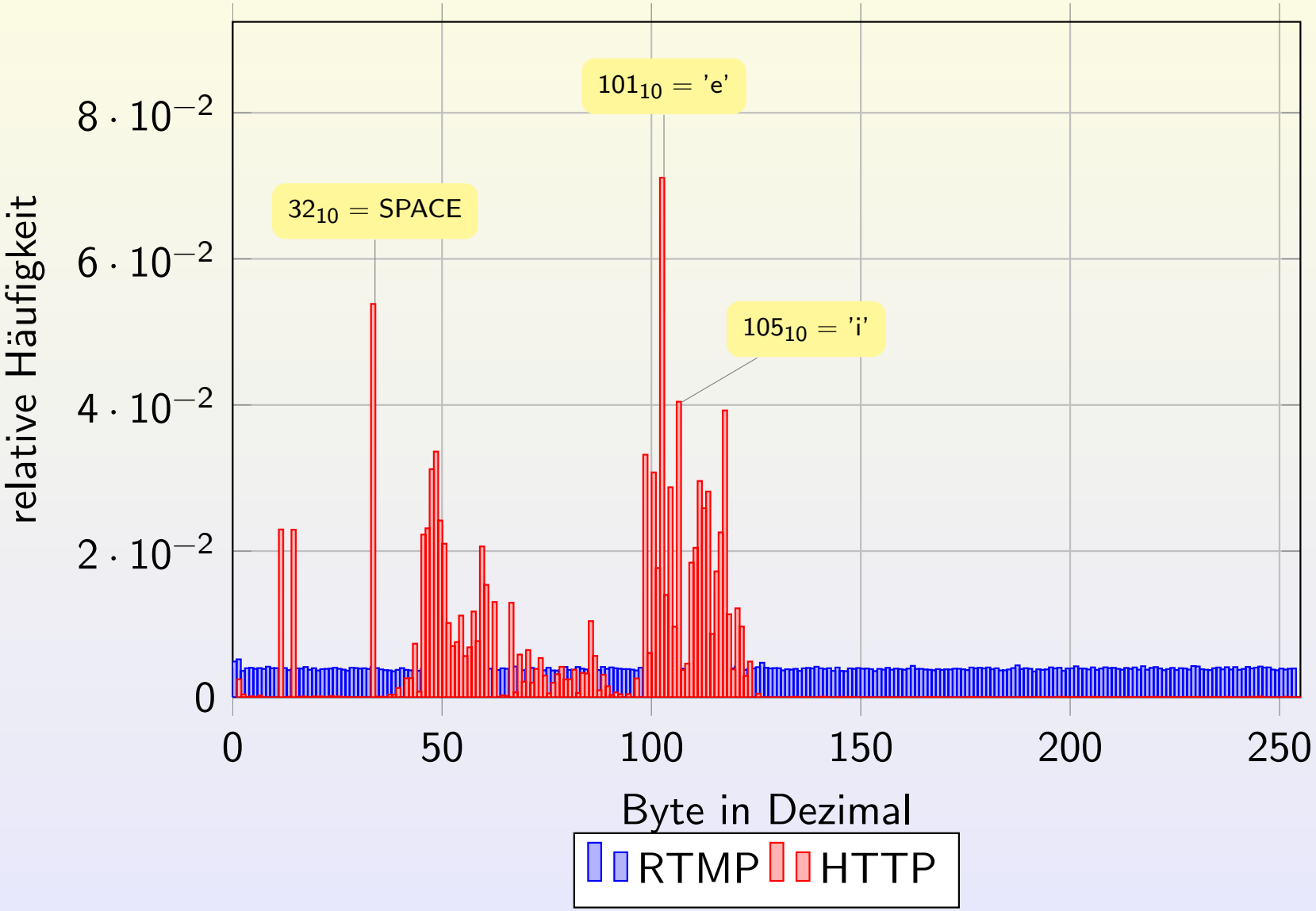
- auch Häufigkeitsanalyse genannt
- operiert auf dem ersten TCP-Paket in jede Richtung
- zählt die relative Häufigkeit eines Bytes in Bezug auf seine Größe

Beispiel

HTTP

```
47 45 54 20    2F 66 6F 6F    G E T    / f o o
48 54 54 50    2F 31 2E 31    H T T P / 1 . 1
```

Messverfahren: Byte-Frequenz



Messverfahren: Richtungswechsel

- misst die Anzahl der Richtungswechsel
- bei n Pakten sind $(n-1)$ Richtungswechsel möglich
- Differenzierung zwischen:
  - häufige Richtungswechsel → Interaktive Protokolle
  - seltene Richtungswechsel → Streaming Protokolle

Messverfahren: Richtungswechsel

- misst die Anzahl der Richtungswechsel
- bei n Pakten sind $(n-1)$ Richtungswechsel möglich
- Differenzierung zwischen:
  - häufige Richtungswechsel → Interaktive Protokolle
  - seltene Richtungswechsel → Streaming Protokolle

Messverfahren: Richtungswechsel

- misst die Anzahl der Richtungswechsel
- bei n Pakten sind $(n-1)$ Richtungswechsel möglich
- Differenzierung zwischen:
  - häufige Richtungswechsel $\rightarrow$ Interaktive Protokolle
  - seltene Richtungswechsel $\rightarrow$ Streaming Protokolle

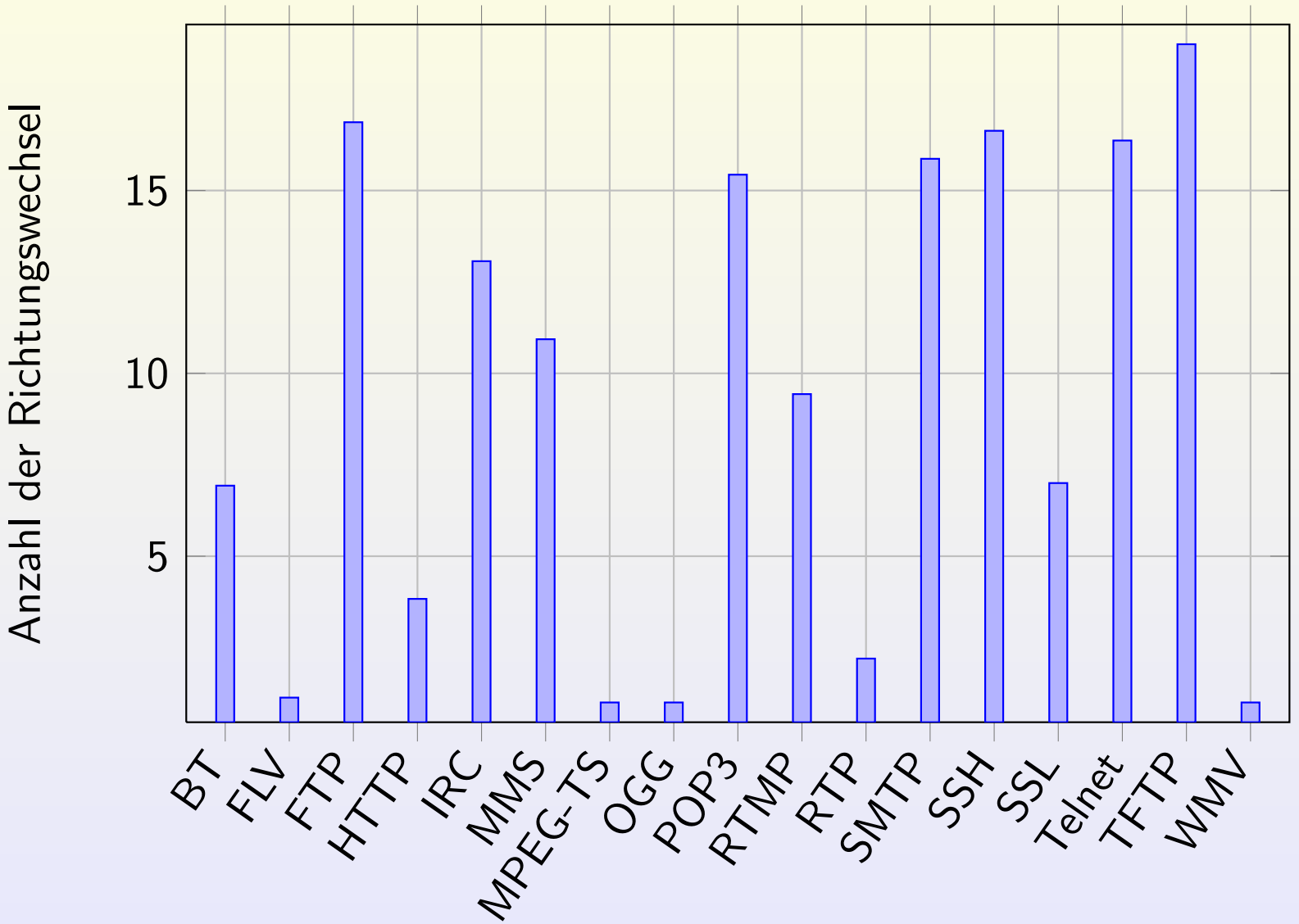
Messverfahren: Richtungswechsel

- misst die Anzahl der Richtungswechsel
- bei n Pakten sind $(n-1)$ Richtungswechsel möglich
- Differenzierung zwischen:
  - häufige Richtungswechsel $\rightarrow$ Interaktive Protokolle
  - seltene Richtungswechsel $\rightarrow$ Streaming Protokolle

Messverfahren: Richtungswechsel

- misst die Anzahl der Richtungswechsel
- bei n Pakten sind $(n-1)$ Richtungswechsel möglich
- Differenzierung zwischen:
  - häufige Richtungswechsel $\rightarrow$ Interaktive Protokolle
  - seltene Richtungswechsel $\rightarrow$ Streaming Protokolle

Messverfahren: Richtungswechsel



Messverfahren: Datenmenge pro Richtung

- misst die Datenmenge in jede Richtung
- Verfahren setzt Up- und Download in prozentuales Verhältnis
- Differenzierung zwischen:
  - Up/Down ausbalaciert
  - fast nur Download
  - fast nur Upload

Messverfahren: Datenmenge pro Richtung

- misst die Datenmenge in jede Richtung
- Verfahren setzt Up- und Download in prozentuales Verhältnis
- Differenzierung zwischen:
  - Up/Down ausbalaciert
  - fast nur Download
  - fast nur Upload

Messverfahren: Datenmenge pro Richtung

- misst die Datenmenge in jede Richtung
- Verfahren setzt Up- und Download in prozentuales Verhältnis
- Differenzierung zwischen:
  - Up/Down ausbalaciert
  - fast nur Download
  - fast nur Upload

Messverfahren: Datenmenge pro Richtung

- misst die Datenmenge in jede Richtung
- Verfahren setzt Up- und Download in prozentuales Verhältnis
- Differenzierung zwischen:
  - Up/Down ausbalaciert
  - fast nur Download
  - fast nur Upload

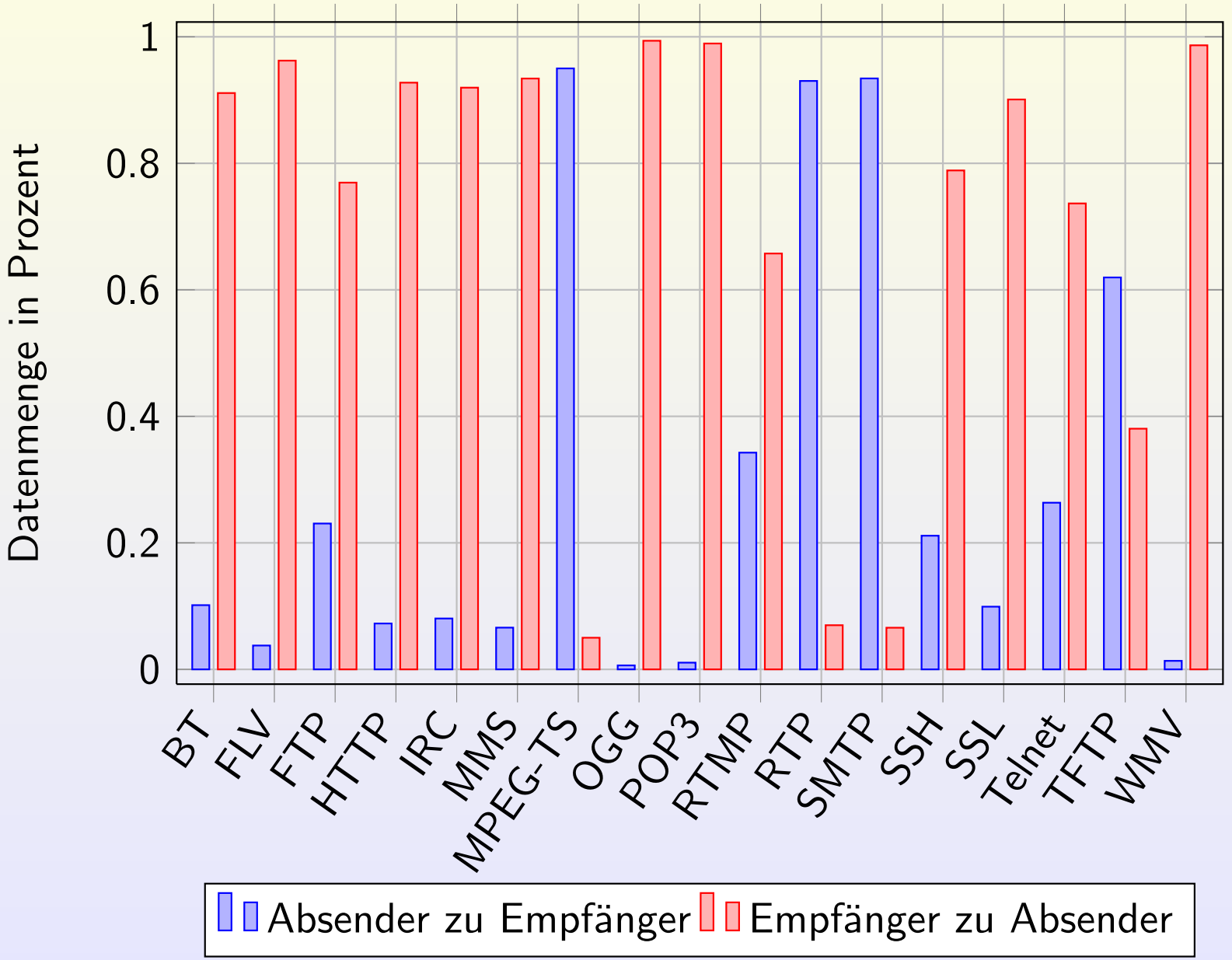
Messverfahren: Datenmenge pro Richtung

- misst die Datenmenge in jede Richtung
- Verfahren setzt Up- und Download in prozentuales Verhältnis
- Differenzierung zwischen:
  - Up/Down ausbalaciert
  - fast nur Download
  - fast nur Upload

Messverfahren: Datenmenge pro Richtung

- misst die Datenmenge in jede Richtung
- Verfahren setzt Up- und Download in prozentuales Verhältnis
- Differenzierung zwischen:
  - Up/Down ausbalaciert
  - fast nur Download
  - fast nur Upload

Messverfahren: Datenmenge pro Richtung



Messverfahren: Paketgröße

- misst die Paketgröße des ersten Pakets
- Informationen über die Initialisierung
- Oft gibt es eine minimale und maximale Paketgröße

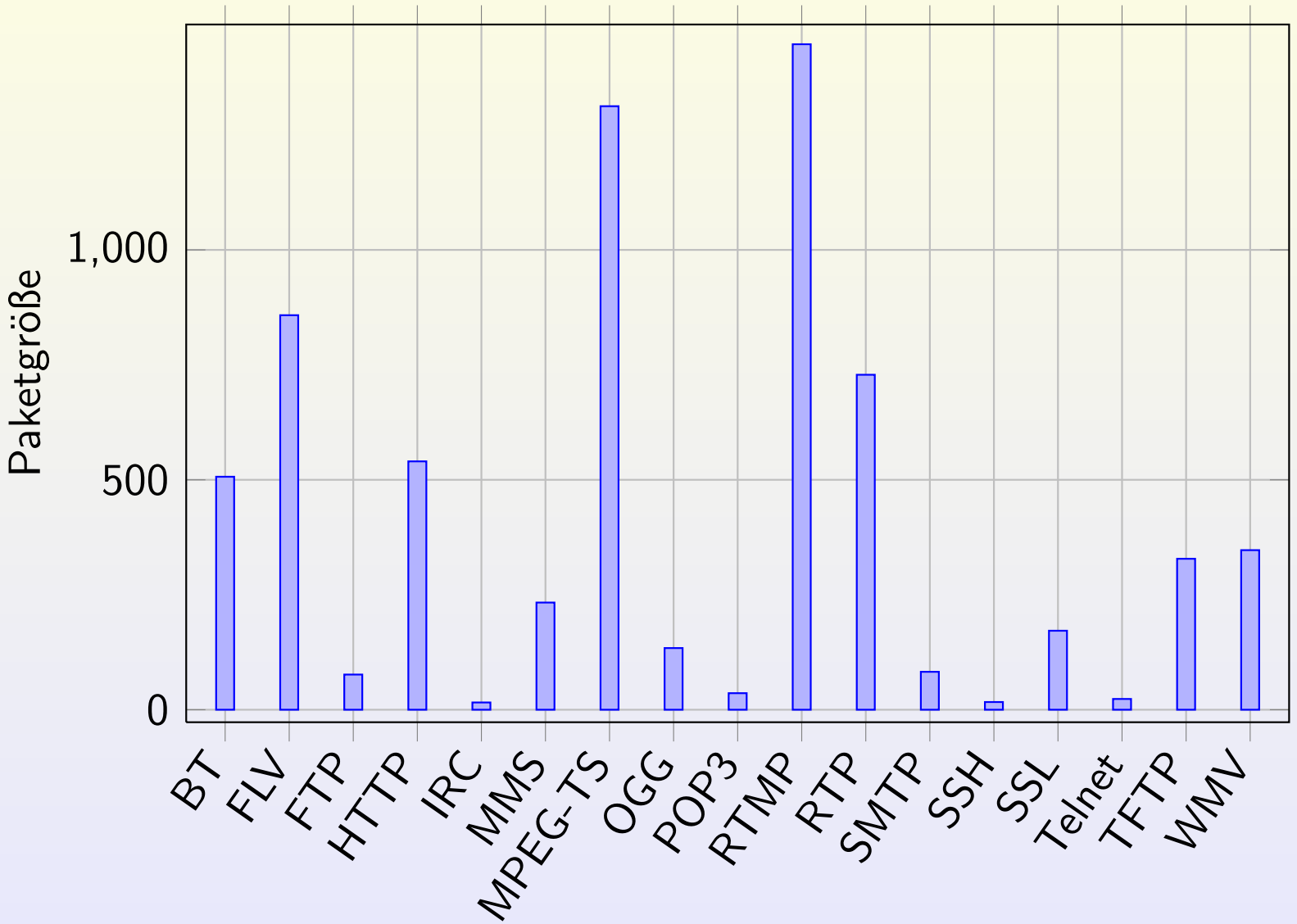
Messverfahren: Paketgröße

- misst die Paketgröße des ersten Pakets
- Informationen über die Initialisierung
- Oft gibt es eine minimale und maximale Paketgröße

Messverfahren: Paketgröße

- misst die Paketgröße des ersten Pakets
- Informationen über die Initialisierung
- Oft gibt es eine minimale und maximale Paketgröße

Messverfahren: Paketgröße



Restliche Messverfahren

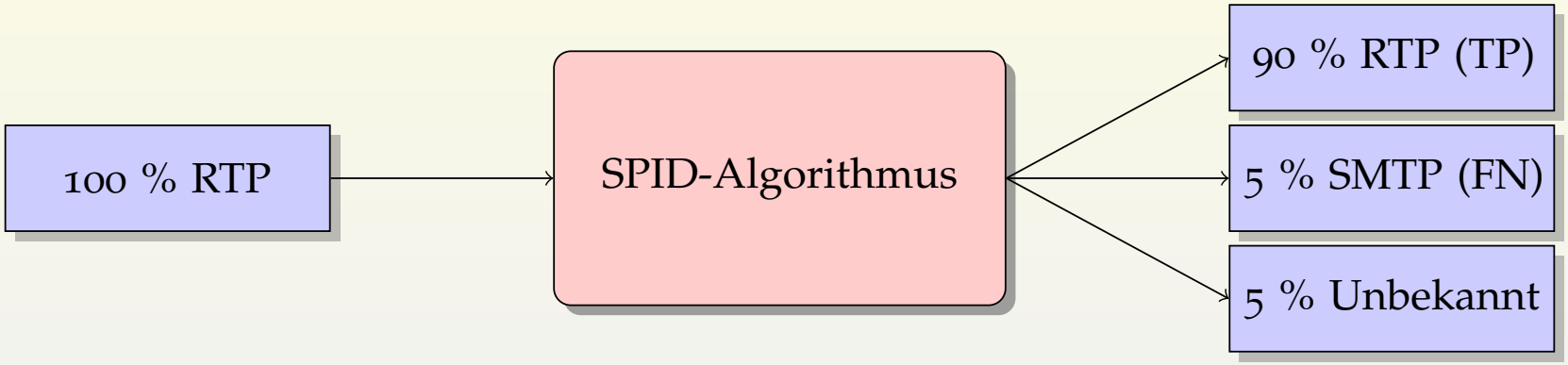
- Byte-Frequenz
- Anzahl der Richtungswechsel
- Datenmenge pro Richtung
- Hashfunktion über die ersten 4 Bytes
- Aktion und Reaktion
- Entropie
- Periodizität von Byte-Paaren
- Similarität der ersten drei Bytes
- Unicode-Frequenz
- Bit-Frequenz
- Payload-Größe des ersten Pakets

Restliche Messverfahren

- Byte-Frequenz
- Anzahl der Richtungswechsel
- Datenmenge pro Richtung
- Hashfunktion über die ersten 4 Bytes
- Aktion und Reaktion
- Entropie
- Periodizität von Byte-Paaren
- Similarität der ersten drei Bytes
- Unicode-Frequenz
- Bit-Frequenz
- Payload-Größe des ersten Pakets

Inhaltsverzeichnis

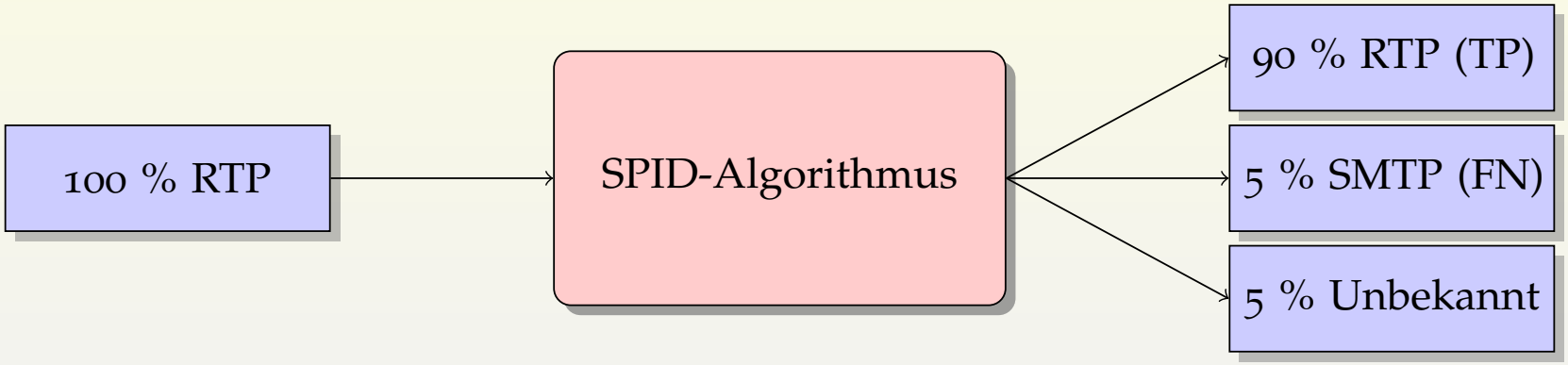
Evaluierungsmethoden



Kontigenztabelle

	tatsächlich positiv	tatsächlich negativ
positiv vorhergesagt	richtig positiv (TP)	falsch positiv (FP)
negativ vorhergesagt	falsch negativ (FN)	richtig negativ (TN)

Evaluierungsmethoden



Kontigenztabelle

	tatsächlich positiv	tatsächlich negativ
positiv vorhergesagt	richtig positiv (TP)	falsch positiv (FP)
negativ vorhergesagt	falsch negativ (FN)	richtig negativ (TN)

Evaluierungsmethoden

Trefferquote

ist die Wahrscheinlichkeit mit der ein Protokoll richtig erkannt wird

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Genauigkeit

ist die Wahrscheinlichkeit der Treffermenge

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

F-Maß

kombiniert Genauigkeit und Trefferquote

$$\text{F} - \text{Maß} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Evaluierungsmethoden

Trefferquote

ist die Wahrscheinlichkeit mit der ein Protokoll richtig erkannt wird

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Genauigkeit

ist die Wahrscheinlichkeit der Treffermenge

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

F-Maß

kombiniert Genauigkeit und Trefferquote

$$\text{F} - \text{Maß} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Evaluierungsmethoden

Trefferquote

ist die Wahrscheinlichkeit mit der ein Protokoll richtig erkannt wird

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Genauigkeit

ist die Wahrscheinlichkeit der Treffermenge

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

F-Maß

kombiniert Genauigkeit und Trefferquote

$$\text{F} - \text{Maß} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Datenbestand

- Beschaffung aus freien Quellen
  - http://www.openpacket.org
  - http://www.pcapr.net
  - http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände ▸ Kontakt

Datenbestand

- Beschaffung aus freien Quellen
  - `http://www.openpacket.org`
  - `http://www.pcapr.net`
  - `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html`
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände ▸ Kontakt

Datenbestand

- Beschaffung aus freien Quellen
  - `http://www.openpacket.org`
  - `http://www.pcapr.net`
  - `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html`
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände ▸ Kontakt

Datenbestand

- Beschaffung aus freien Quellen
  - `http://www.openpacket.org`
  - `http://www.pcapr.net`
  - `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html`
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände ▸ Kontakt

Datenbestand

- Beschaffung aus freien Quellen
  - `http://www.openpacket.org`
  - `http://www.pcapr.net`
  - `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html`
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände ▸ Kontakt

Datenbestand

- Beschaffung aus freien Quellen
  - `http://www.openpacket.org`
  - `http://www.pcapr.net`
  - `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html`
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände

Kontakt

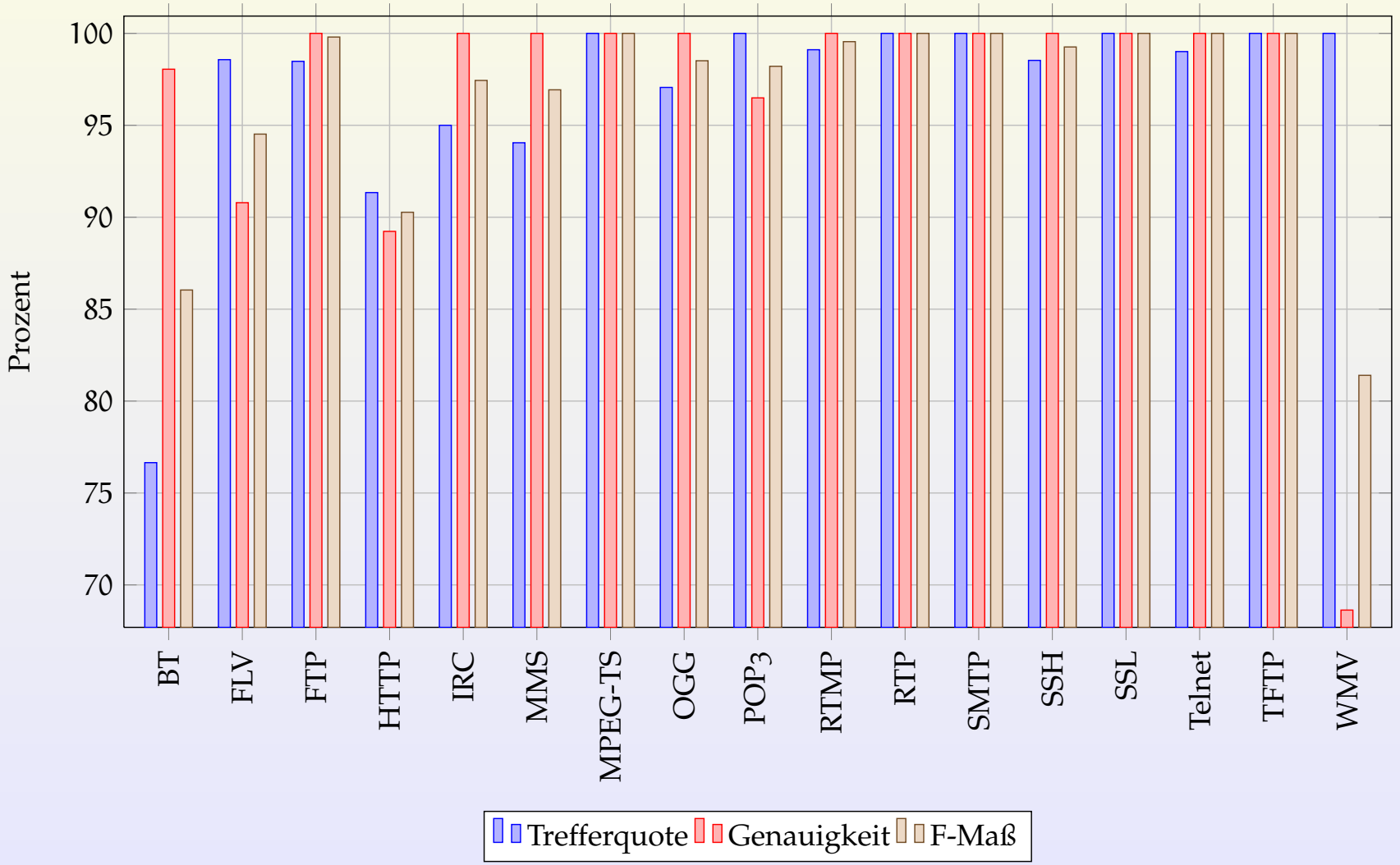
Datenbestand

- Beschaffung aus freien Quellen
  - `http://www.openpacket.org`
  - `http://www.pcapr.net`
  - `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html`
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände Kontakt

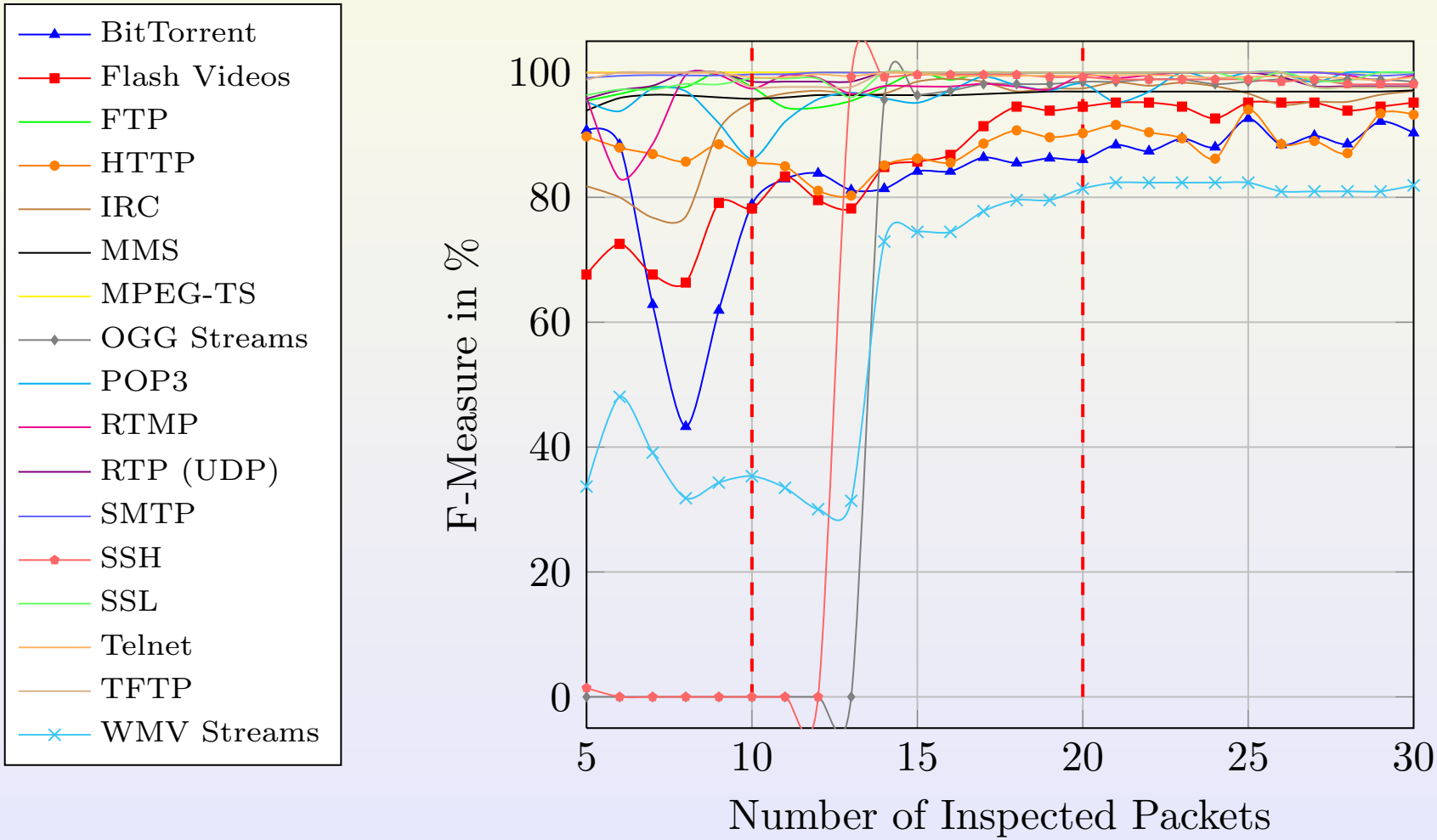
Datenbestand

- Beschaffung aus freien Quellen
  - `http://www.openpacket.org`
  - `http://www.pcapr.net`
  - `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html`
- Erstellung eigener Mitschnitte
  - soviel wie möglich automatisiert mit Perl Scripten
- 3135 Flows für 17 Protokolle die 1.5 GB ergeben
- Ich bin dankbar für Hinweise auf Datenbestände ▸ Kontakt

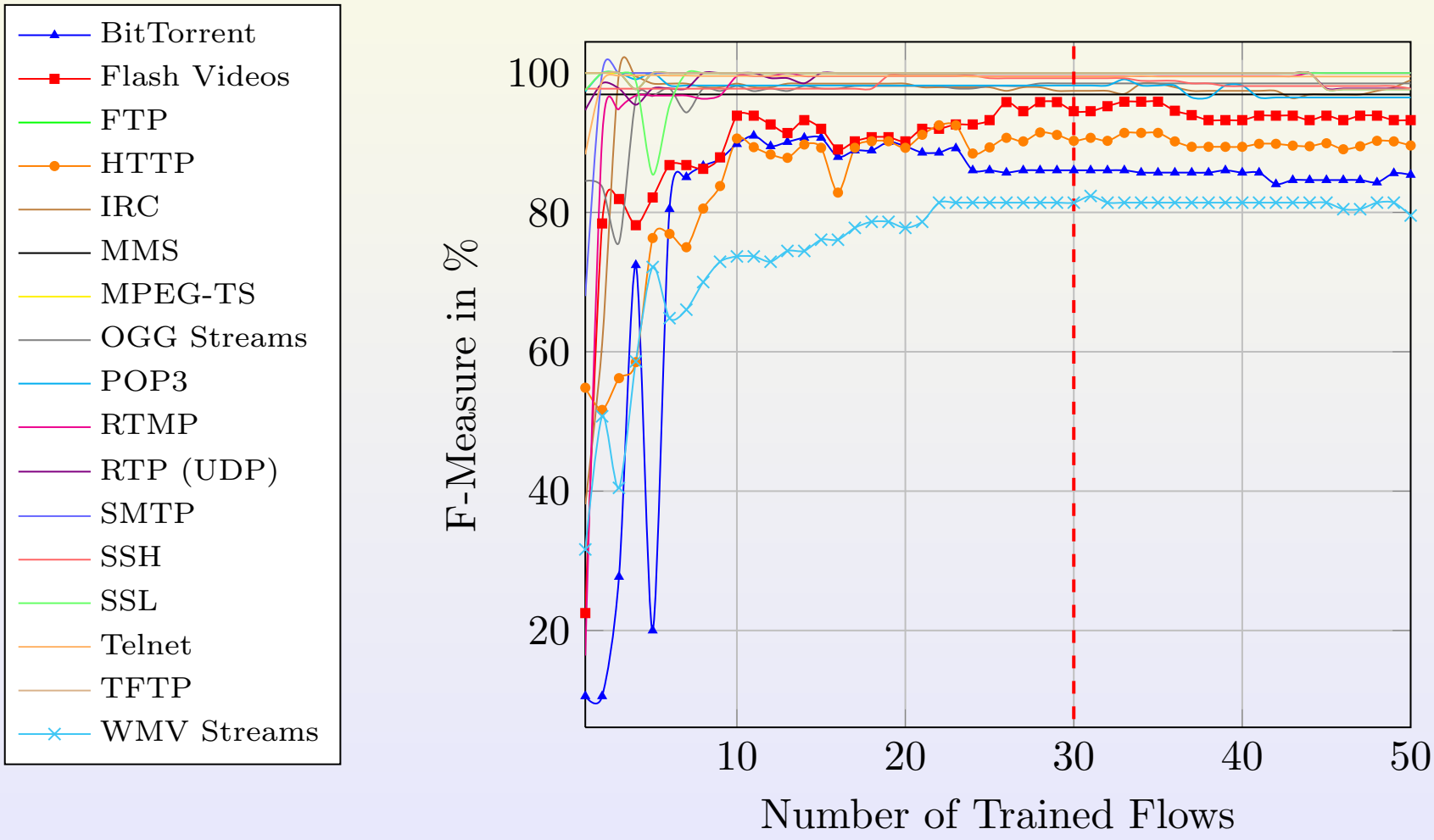
Ergebnisse


Prozent
100
95
90
85
80
75
70
BT
FLV
FTP
HTTP
IRC
MMS
MPEG-TS
OGG
POP3
RTMP
RTP
SMTP
SSH
SSL
Telnet
TFTP
WMV
Trefferquote
Genauigkeit
F-Maß

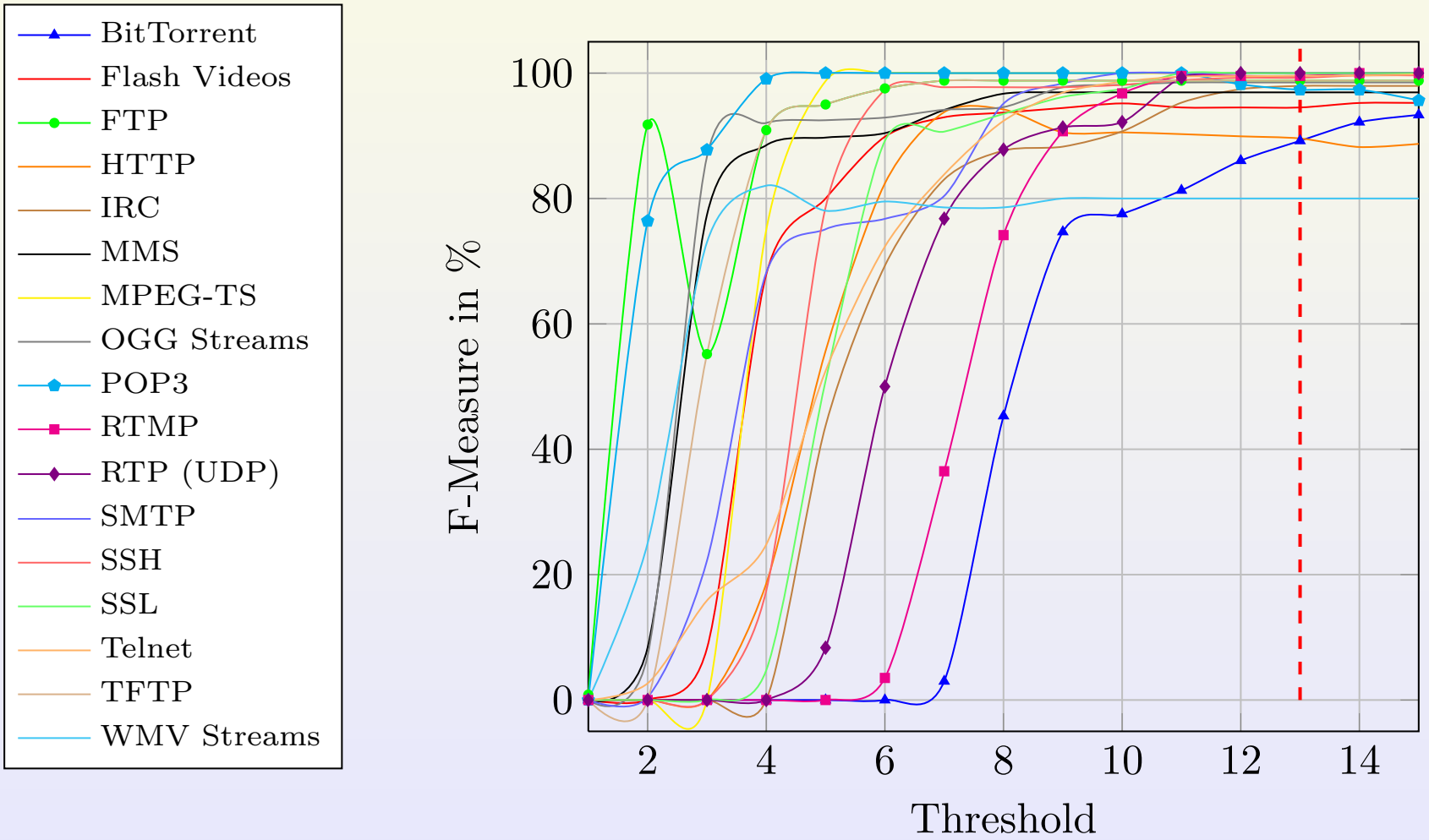
Ergebnisse


BitTorrent
Flash Videos
FTP
HTTP
IRC
MMS
MPEG-TS
OGG Streams
POP3
RTMP
RTP (UDP)
SMTP
SSH
SSL
Telnet
TFTP
WMV Streams
F-Measure in %
100
80
60
40
20
0
5
10
15
20
25
30
Number of Inspected Packets

Ergebnisse


BitTorrent
Flash Videos
FTP
HTTP
IRC
MMS
MPEG-TS
OGG Streams
POP3
RTMP
RTP (UDP)
SMTP
SSH
SSL
Telnet
TFTP
WMV Streams
F-Measure in %
100
80
60
40
20
10
20
30
40
50
Number of Trained Flows

Ergebnisse


BitTorrent
Flash Videos
FTP
HTTP
IRC
MMS
MPEG-TS
OGG Streams
POP3
RTMP
RTP (UDP)
SMTP
SSH
SSL
Telnet
TFTP
WMV Streams
F-Measure in %
100
80
60
40
20
0
2
4
6
8
10
12
14
Threshold

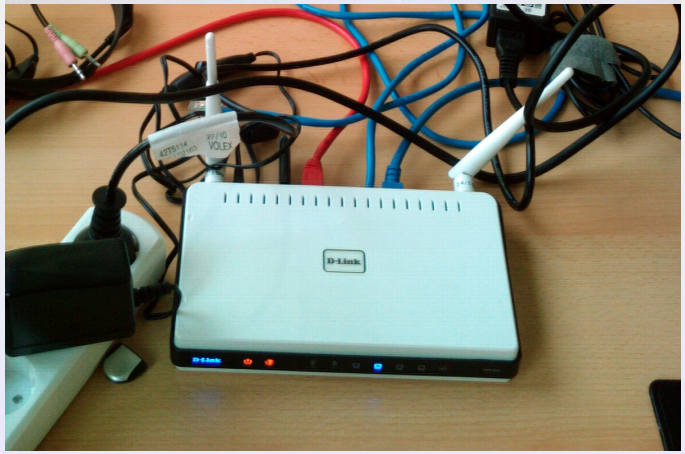
Geschwindigkeitstest

Linksys WRT54GL

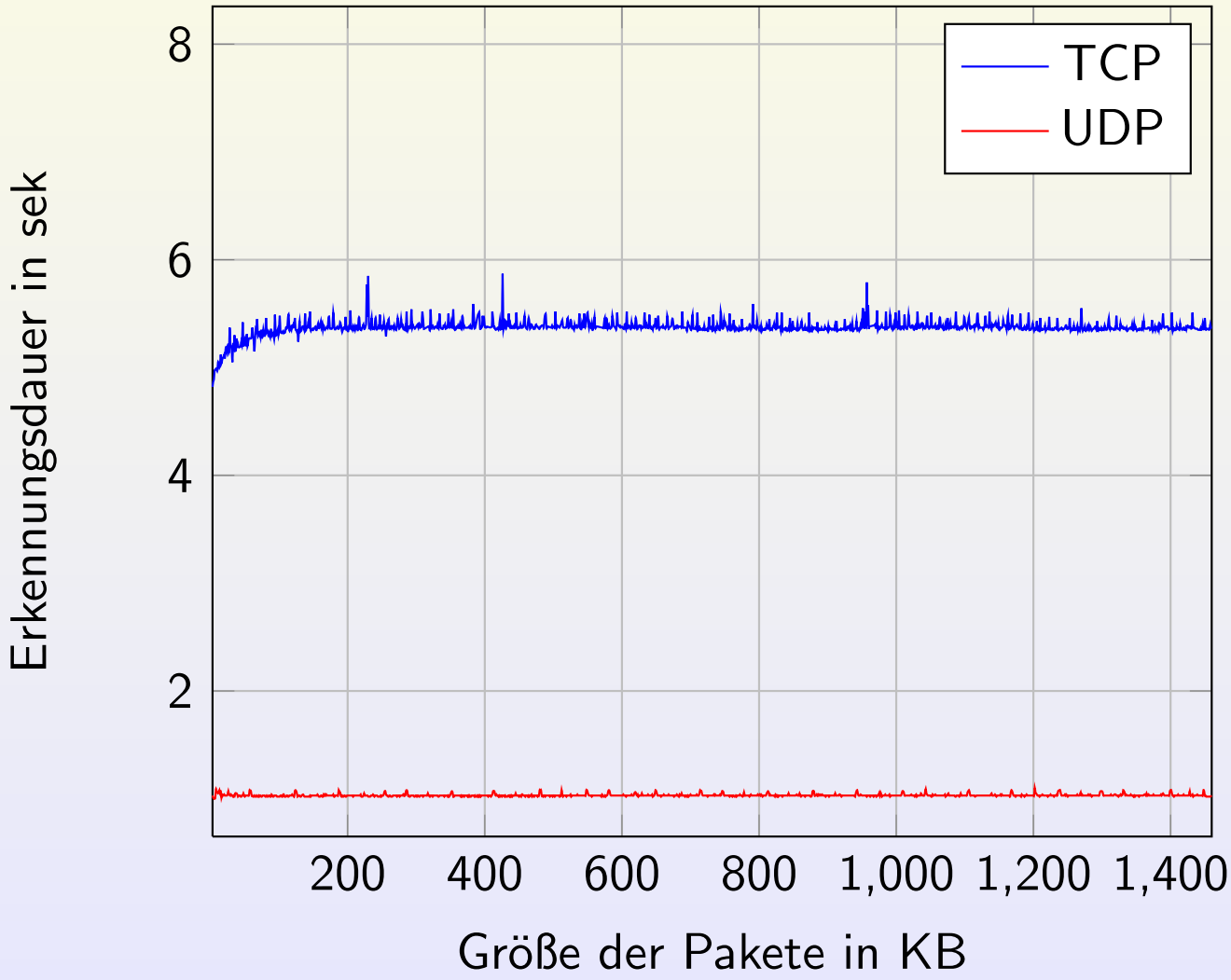
- OS: OpenWRT 8.09 Kamikaze
- Kernel: 2.4.35.4
- CPU: 200 MHz MIPSel
- RAM: 16 MiB

Dlink DIR 825

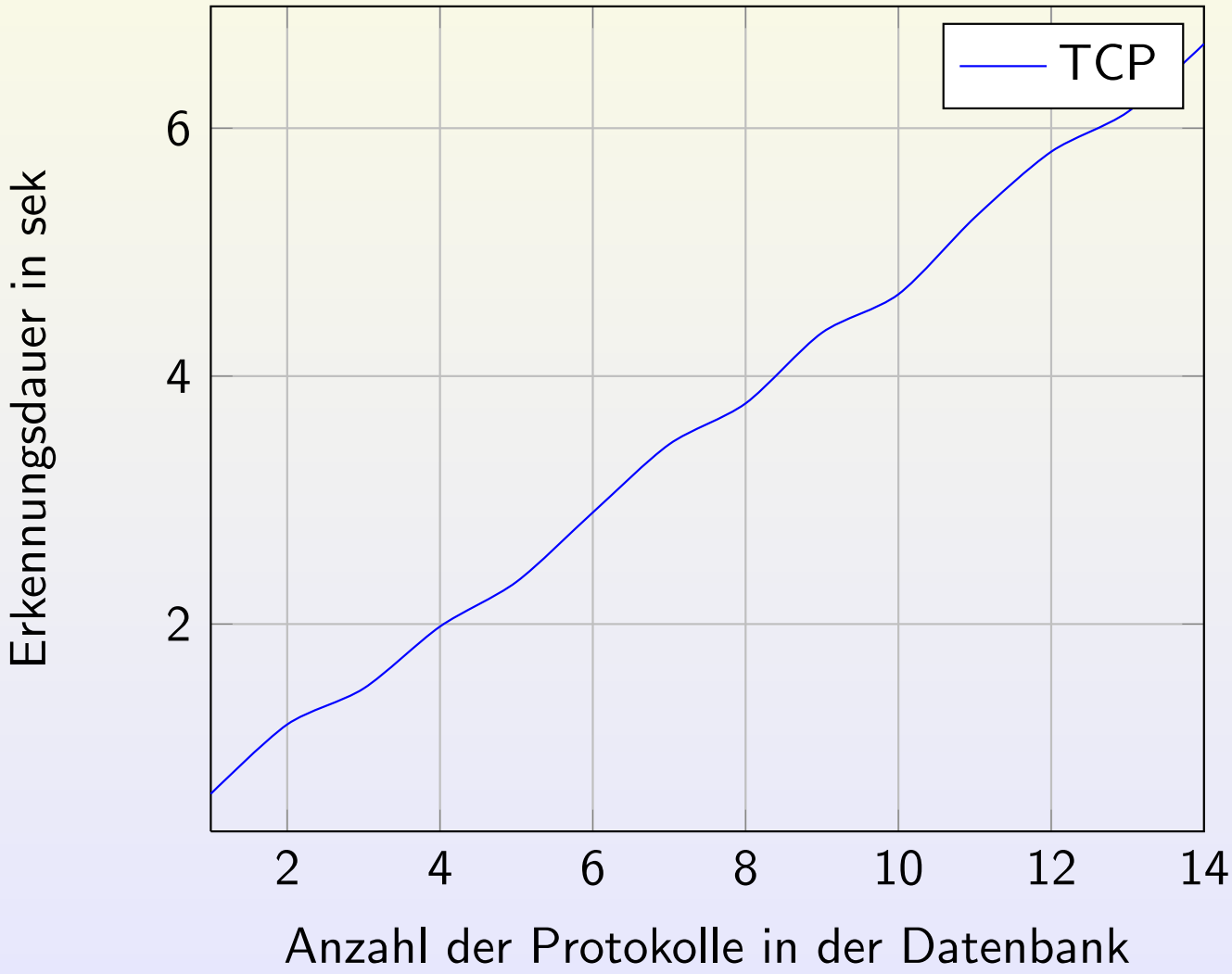
- OS: OpenWRT Backfire
- Kernel: 2.6.32-16
- CPU: 680 MHz MIPS
- RAM: 64 MiB



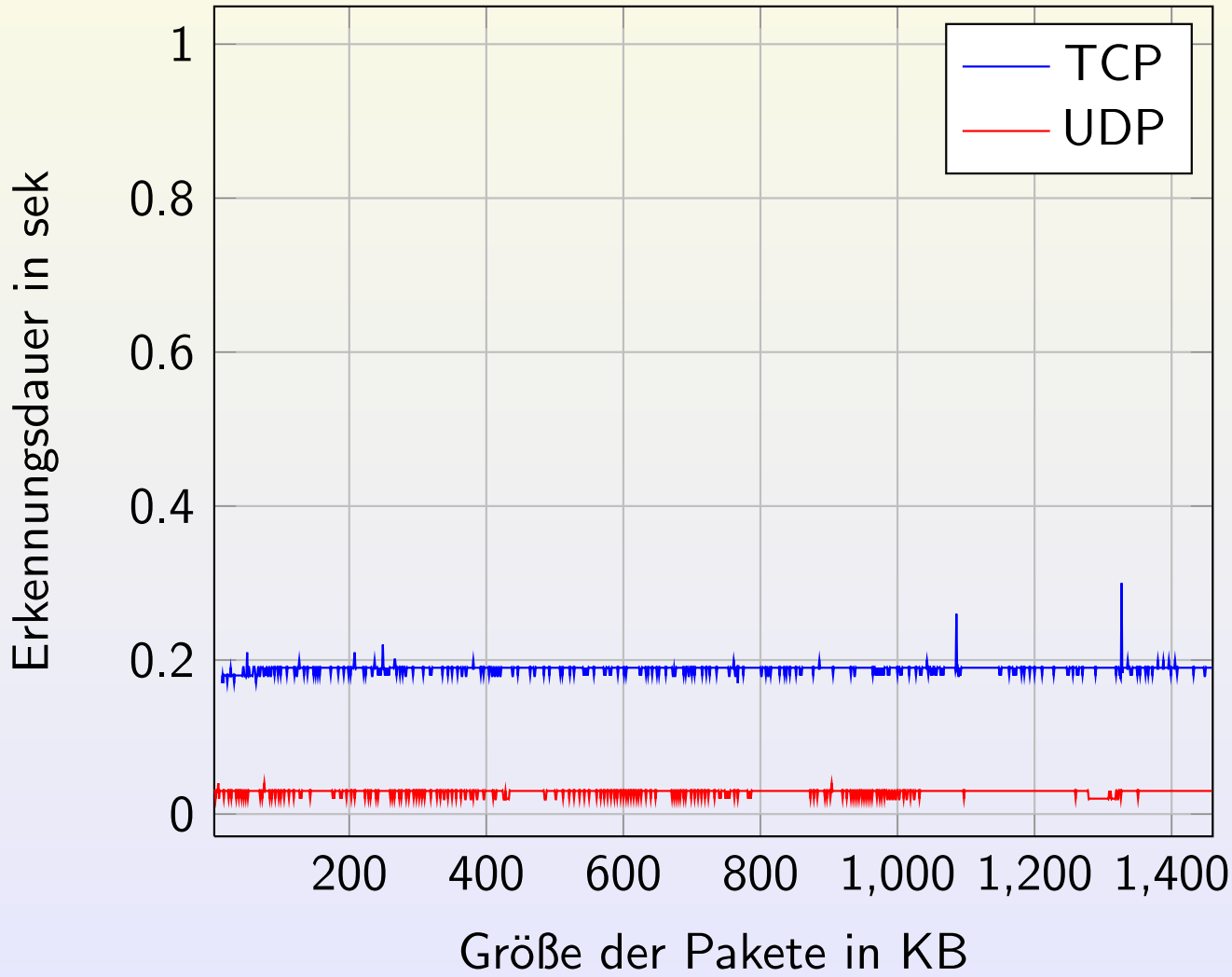
Linksys WRT54GL 1/2



Linksys WRT54GL 2/2



Dlink DIR 825



Inhaltsverzeichnis

Protokoll-Obfuskation

Definition

Protokoll-Obfuskation hat das Ziel das Protkoll zu verschleiern und so eine Identifizierung zu vermeiden.

Arten

- Payload Obfuskation
  - Verschlüsselung (z.B. VPN)
- Flow-Level Obfuskation

Protokolle

- BitTorrent (MSE - Message Stream Encryption)
- eMule
- Skype
- ...

Protokoll-Obfuskation

Definition

Protokoll-Obfuskation hat das Ziel das Protkoll zu verschleiern und so eine Identifizierung zu vermeiden.

Arten

- Payload Obfuskation
  - Verschlüsselung (z.B. VPN)
- Flow-Level Obfuskation

Protokolle

- BitTorrent (MSE - Message Stream Encryption)
- eMule
- Skype
- ...

Protokoll-Obfuskation

Definition

Protokoll-Obfuskation hat das Ziel das Protkoll zu verschleiern und so eine Identifizierung zu vermeiden.

Arten

- Payload Obfuskation
  - Verschlüsselung (z.B. VPN)
- Flow-Level Obfuskation

Protokolle

- BitTorrent (MSE - Message Stream Encryption)
- eMule
- Skype
- ...

Protokoll-Obfuskation

Definition

Protokoll-Obfuskation hat das Ziel das Protkoll zu verschleiern und so eine Identifizierung zu vermeiden.

Arten

- Payload Obfuskation
  - Verschlüsselung (z.B. VPN)
- Flow-Level Obfuskation

Protokolle

- BitTorrent (MSE - Message Stream Encryption)
- eMule
- Skype
- ...

Protokoll-Obfuskation

Definition

Protokoll-Obfuskation hat das Ziel das Protkoll zu verschleiern und so eine Identifizierung zu vermeiden.

Arten

- Payload Obfuskation
  - Verschlüsselung (z.B. VPN)
- Flow-Level Obfuskation

Protokolle

- BitTorrent (MSE - Message Stream Encryption)
- eMule
- Skype
- ...

Protokoll-Obfuskation

Definition

Protokoll-Obfuskation hat das Ziel das Protkoll zu verschleiern und so eine Identifizierung zu vermeiden.

Arten

- Payload Obfuskation
  - Verschlüsselung (z.B. VPN)
- Flow-Level Obfuskation

Protokolle

- BitTorrent (MSE - Message Stream Encryption)
- eMule
- Skype
- ...

Protokoll-Obfuskation

Definition

Protokoll-Obfuskation hat das Ziel das Protkoll zu verschleiern und so eine Identifizierung zu vermeiden.

Arten

- Payload Obfuskation
  - Verschlüsselung (z.B. VPN)
- Flow-Level Obfuskation

Protokolle

- BitTorrent (MSE - Message Stream Encryption)
- eMule
- Skype
- ...

Obfuskation verbessern

- nur Payload Verschlüsselung reicht nicht
- Werte die zufälliger sein sollten:
  - Senderichtung der Pakete
  - Inter-Arrival Time
  - Paketgröße
- Alternativen
  - Protokolle in andere Protokolle verstecken

Obfuskation verbessern

- nur Payload Verschlüsselung reicht nicht
- Werte die zufälliger sein sollten:
  - Senderichtung der Pakete
  - Inter-Arrival Time
  - Paketgröße
- Alternativen
  - Protokolle in andere Protokolle verstecken

Obfuskation verbessern

- nur Payload Verschlüsselung reicht nicht
- Werte die zufälliger sein sollten:
  - Senderichtung der Pakete
  - Inter-Arrival Time
  - Paketgröße
- Alternativen
  - Protokolle in andere Protokolle verstecken

Obfuskation verbessern

- nur Payload Verschlüsselung reicht nicht
- Werte die zufälliger sein sollten:
  - Senderichtung der Pakete
  - Inter-Arrival Time
  - Paketgröße
- Alternativen
  - Protokolle in andere Protokolle verstecken

Obfuskation verbessern

- nur Payload Verschlüsselung reicht nicht
- Werte die zufälliger sein sollten:
  - Senderichtung der Pakete
  - Inter-Arrival Time
  - Paketgröße
- Alternativen
  - Protokolle in andere Protokolle verstecken

Obfuskation verbessern

- nur Payload Verschlüsselung reicht nicht
- Werte die zufälliger sein sollten:
  - Senderichtung der Pakete
  - Inter-Arrival Time
  - Paketgröße
- Alternativen
  - Protokolle in andere Protokolle verstecken

Obfuskation verbessern

- nur Payload Verschlüsselung reicht nicht
- Werte die zufälliger sein sollten:
  - Senderichtung der Pakete
  - Inter-Arrival Time
  - Paketgröße
- Alternativen
  - Protokolle in andere Protokolle verstecken

Ende

Danke für eure Aufmerksamkeit! ☺

Gibt es noch Fragen?

Kontakt

E-Mail: `florian-27c3@adamsky.it`

Twitter: `@c1t`