



**TECHNISCHE  
UNIVERSITÄT  
DRESDEN**

---

Faculty of Computer Science Institute of Systems Architecture, Chair of Privacy and Data Security

---

Bachelor-Arbeit

# Thesis title

Student name

Geboren am: 25. Dezember 1999 in Dresden

Matrikelnummer: 4803900

Immatrikulationsjahr: 2018

Betreuer

**Supervisor**

Betreuender Hochschullehrer

**Prof. Dr. Professor**



## **AUFGABENSTELLUNG FÜR DIE BACHELORARBEIT**

Name, Vorname des Studenten: Köllner, Tobias  
Immatrikulationsnummer: 4803900  
Studiengang: Bachelor Informatik  
E-Mail: tobias.koellner@mailbox.tu-dresden.de

Thema: Kontextsensitivität für Netzwerk-Sicherheits-Monitoring

### *Zielstellung:*

Zur Erhöhung der Sicherheit von Netzwerken werden häufig Techniken wie Network security monitoring (NSM) sowie intrusion detection (IDS) verwendet. Insbesondere die Menge an falsch-positiven Meldungen stellt hierbei eine große Herausforderung dar. Um dieses Problem sowie die generelle Interpretierbarkeit der Beobachtungsdaten zu verbessern können zusätzliche Informationen gesammelt werden, die man als Netzwerkkontext bezeichnen kann.

Der erste Teil der Arbeit soll untersuchen, wie solche Kontextinformationen sowohl während des Betriebs von NSM-Systemen als auch während der Signatur/Modell-Erstellung im Rahmen von IDS gesammelt und verwendet werden können. Ziele hierbei sind, solche Möglichkeiten auf theoretischer Ebene wie auch von open-source Implementierungen zu sichten und hierbei einen kritischen Kontextbegriff im Vergleich zu etablierten Kontextbegriffen in der Literatur in Bezug auf Netzwerke zu entwickeln.

Aufbauend auf diesen Resultaten kann eine einfache praktische Auswertung verfügbarer Implementierungen mit explizitem Hinblick auf die kontextsensitiven Fähigkeiten durchgeführt werden. Hierfür ist ein Plan für eine minimale Testumgebung sowie das Design von geeignetem Netzwerkverkehr erforderlich.

### *In der Arbeit sollen schwerpunktmäßig folgende Teilaufgaben bearbeitet werden:*

- Überblick und Analyse existierender NSM-Ansätze in Bezug auf Kontext- und Umgebungsbewusstsein. Hierbei sollen mindestens folgende Forschungsfragen bearbeitet werden:
  - Lassen sich generelle Kategorien von Kontext im Rahmen von NSM formulieren?
  - Welche Kontextinformation kann während der Regelgenerierung im Unterschied zu generell verfügbaren Regeln aus öffentlichen Quellen berücksichtigt werden?
  - Welche Techniken wurden vorgeschlagen, um Kontextinformation während des regulären Betriebs zu sammeln und wie werden diese dann genutzt?
  - Inwiefern gibt es Möglichkeiten der Anpassungen an Änderungen der Netzwerkkumgebung, etwa der Topologie oder der genutzten Protokolle?
  - Wie stark trägt Kontextinformation zur Reduktion von falsch-Positiven bei?
- Es soll eine Evaluation der Fähigkeit von Kontextnutzung von verfügbaren open-source-Implementierungen (etwa Plugins in Zeek oder Netzdefinitionen in Suricata) durchgeführt werden.

Verantwortlicher Hochschullehrer:  
Institut:  
Beginn am: 3.6.2022

Dr. Stefan Köpsell  
Systemarchitektur  
Einzureichen am: 19.8. 2022

3.6.22, T Köllner  
.....  
Datum, Unterschrift der/des Studierenden

.....  
Unterschrift des betreuenden Hochschullehrers

### Selbstständigkeitserklärung

Hiermit versichere ich, dass ich das vorliegende Dokument mit dem Titel *Thesis title* selbstständig und ohne unzulässige Hilfe Dritter verfasst habe. Es wurden keine anderen als die in diesem Dokument angegebenen Hilfsmittel und Quellen benutzt. Die wörtlichen und sinngemäß übernommenen Zitate habe ich als solche kenntlich gemacht. Während der Anfertigung dieses Dokumentes wurde ich nur von folgenden Personen unterstützt:

Supervisor name

Zusätzliche Personen waren an der geistigen Herstellung des vorliegenden Dokumentes nicht beteiligt. Mir ist bekannt, dass die Nichteinhaltung dieser Erklärung zum nachträglichen Entzug des Hochschulabschlusses führen kann.

Student name

## **Zusammenfassung**

The abstract serves as a summary of the work. It should concisely (around half a page) answer the following questions:

- What is the problem tackled in this thesis?
- Why is this problem relevant?
- What contributions does this thesis contain with regards to the problem?
- Which scientific method(s) was/were used and what were their results?

# Inhaltsverzeichnis

Zusammenfassung	4
1 Einführung	10
2 Hintergrund	11
2.1 Kontextbegriff	11
2.1.1 Historische Entwicklung	11
2.1.2 Zugriffskontrolle	12
2.2 Kontextkategorisierung	12
2.2.1 Analyse bereits vorgeschlagener Kategorien	12
3 Voraussetzungen und verwandte Arbeiten	14
4 Design	15
4.1 Notwendigkeit einer Taxonomie	15
4.2 Erstellung einer Taxonomie	15
4.2.1 Mensch	15
4.2.2 Netzwerk	17
4.2.3 Historie	19
4.3 Form des Kontextes	20
4.3.1 Historie	21
4.4 Kontextgewinnung	21
4.5 Umwandlung der Taxonomie in IDS-Signaturen	21
4.5.1 Entscheidung für ein IDS	22
4.5.2 Verbesserung der Schwächen	23
4.6 Anspruch an das IDS	24
4.6.1 Einordnung der Kriterien	25
4.6.2 Auswahl der Kriterien	25
5 Implementierung	26
5.1 Erläuterung der wichtigsten Komponenten	26
5.1.1 Zeek	26
5.1.2 Zeek-Agent	26
5.2 Versuchsaufbau	27
5.2.1 Erzeugung	27
5.2.2 Mitschnitt	27
5.2.3 Logging	27

5.3	Skripte . . . . .	28
5.3.1	Geografische Koordinaten und Ortszeit . . . . .	28
5.3.2	Verwendete Ports . . . . .	29
5.3.3	DNS-Auflösung . . . . .	29
<b>6</b>	<b>Evaluation</b>	<b>31</b>
6.1	Mächtigkeit eines kontextsensitiven IDS . . . . .	31
6.2	Vergleich der Kontextkategorien . . . . .	31
6.2.1	Verfügbarkeit der Informationen . . . . .	31
6.2.2	Qualität der Informationen . . . . .	33
6.3	Leistungsverbesserung . . . . .	35
<b>7</b>	<b>Schlussfolgerung und Ausblick</b>	<b>36</b>
7.1	Dateilose Prozesse . . . . .	36

# Abbildungsverzeichnis

4.1	Nutzerzentrierte Kategorisierung . . . . .	17
4.2	Kategorisierung von Kontextinformationen als Netzwerkbestandteile . . . . .	19
4.3	Farblegende der Dynamik und Historie eines Beispieleintrages . . . . .	20
6.1	Placeholder . . . . .	32
6.2	Placeholder . . . . .	33
6.3	Placeholder . . . . .	34
6.4	Placeholder . . . . .	35

# Tabellenverzeichnis

4.1 Placeholder . . . . .	22
---------------------------	----



# Quelltextverzeichnis

5.1	Konfiguration und Versendung eines Pakets . . . . .	27
5.2	Generierung einer Log-Datei mit Verbindungsinformationen . . . . .	28
5.3	Geolokalisierung und Setzen des Grenzwertes . . . . .	28
5.4	Abfrage und Abgleich der Ports . . . . .	29
5.5	Überprüfung der Verbindungsziele eines Endgerätes . . . . .	30

# 1 Einführung

## 2 Hintergrund

### 2.1 Kontextbegriff

Kontext ist ein Begriff unter dem sich die meisten Menschen zwar etwas vorstellen, aber ihn nur schwer erläutern oder gar korrekt und vollständig definieren können [8]. Auch herrscht in verschiedenen Fachbereichen, jeder davon mit anderen Sachverhalten und Problemen die die jeweiligen Autoren versuchen zu lösen, eine andere Auffassung darüber welche Ansprüche eine Definition erfüllen muss. Das erschwert eine allumfassende, konkrete Definition zusätzlich [5]. Ziel der Arbeit ist also nicht Kontext abschließend zu definieren, da dies in Anbetracht der vielen verschiedenen Ansätze und dem fehlenden Konsens wie Kontext zu definieren ist [4, 18] schlicht nicht möglich ist bzw. den Rahmen übersteigen würde. Stattdessen wird versucht, mithilfe relevanter Definitionen, die historische Entwicklung des Kontextbegriffs für den Bereich der Informatik im Allgemeinen und der Zugriffskontrolle im Speziellen darzustellen. Dies soll ein Verständnis dafür schaffen auf welchen Grundlagen, Ansätzen und Ideen der Kontextbegriff dieser Arbeit entstanden und aufgebaut ist. Dies ermöglicht dem Leser eine Einordnung davon wie Kontext und Kontextsensitivität verwendet werden und er kann einen Abgleich mit seinem eigenen Verständnis der Begrifflichkeiten durchführen.

#### 2.1.1 Historische Entwicklung

Die Definition von Kontext ist auch in der Literatur seit jeher ein Thema. Eine der frühesten Definitionen von Kontext von Schilit et al.[16] bestimmt als drei Hauptaspekte von Kontext an welchem Ort, in Gegenwart welcher anderen Personen und in der Nähe welcher Ressourcen sich ein Nutzer befindet.

Des weiteren beinhaltet nach [16] Kontext Attribute wie Beleuchtung, Lautstärke, den Grad der Netzwerkverbindung, Kommunikationskosten, Kommunikationsbandbreite und die soziale Situation.

Nach der Definition von Dey [8] ist "Kontext jede Information die genutzt werden kann um die Situation einer Entität zu charakterisieren. Eine Entität ist eine Person, ein Objekt oder ein Ort mit Relevanz für die Interaktion zwischen Nutzer und Anwendung. Das schließt auch Nutzer und Anwendung selbst mit ein". Sie wird allgemein hin von den meisten anderen Autoren als Quasikonsens akzeptiert [18, 4, 3] oder als Ausgangspunkt für ihre eigene Definition genutzt [21, 11].

Kaltz et al. [19] verstehen Kontext als ein Kontextraum also eine Kombination aus Kontextparametern, Elementen einer domänenspezifischen Ontologie und Dienstleistungsbeschreibungen in Form von  $C = \{U; P; L; T; D; I; S\}$  definiert. Dabei ist  $U$  das Set aus Nutzern und den

dazugehörigen Rollen,  $P$  die Prozesse und Aufgaben,  $L$  der Ort,  $T$  der Zeitfaktor,  $D$  beschreibt das Gerät,  $I$  die verfügbaren Informationen und  $S$  die verfügbaren Dienstleistungen. Ein spezifischer Kontext ist somit ein Punkt in diesem Raum.

Bazire und Brézillon [5] haben 150 Kontextdefinitionen analysiert und sind dabei zu der Erkenntnis gekommen dass Kontext wie eine Begrenzung fungiert welche das Verhalten eines Systems, Nutzers oder Computers in einer bestimmten Tätigkeit beeinflussen. Allerdings herrscht ihrer Ansicht nach kein Konsens darüber ob Kontext extern oder intern, ein Set aus Informationen oder Abläufen, statisch oder dynamisch ist.

Im Rahmen der Zugriffskontrolle Kayes et al.[11] definieren Kontextinformationen in Bezug auf Zugriffskontrollentscheidungen als relevante Informationen über den Zustand einer Entität (Nutzer, Ressource, Ressourcenbesitzer) und deren Umgebung oder die Beziehung zwischen Entitäten.

### 2.1.2 Zugriffskontrolle

## 2.2 Kontextkategorisierung

Nachdem ein Überblick über das Kontextverständnis in der gängigen Literatur gegeben und Kontext im Bezug auf Zugriffskontrolle erläutert wurde folgt eine Vorstellung ausgewählter Ansätze der Kategorisierung.

### 2.2.1 Analyse bereits vorgeschlagener Kategorien

Auch bei der Kategorisierung von Kontext gibt es richtungsweisende Vorschläge. Abowd et al.[2] haben einen der führenden Mechanismen zur Definition von Kontexttypen vorgeschlagen. Sie identifizierten Ort, Zeit, Identität, und Aktivität als primäre Kontexttypen. Weiterhin wird sekundärer Kontext als Kontext definiert der durch Nutzung von Primärkontext erschlossen werden kann. Schilit et al. [16] kategorisieren Kontext basierend auf 3 Fragen, die genutzt werden können um den Kontext zu bestimmen, in 3 Kategorien:

1. Wo man sich befindet: Enthält alle Informationen die sich auf einen Ort beziehen, beispielsweise GPS-Koordinaten, Namen von Institutionen oder Gebäuden (ein Café, ein Krankenhaus, eine Universität), spezifische Namen (z.B.: Technische Universität Dresden) spezifischen Adressen(z.B.: APB Nöthnitzer Str. 46) oder Nutzerpräferenzen (z.B. das Lieblingsrestaurant eines Nutzers)
2. Mit wem man sich zusammen aufhält: Information über die Personen die um einen herum anwesend sind
3. Welche Ressourcen sich in der Nähe befinden: Informationen darüber welche Ressourcen (Maschinen, technische Geräte, Betriebsmittel) sich im direkten Umfeld eines Nutzers befinden

Henricksen et al.[10] ordnet Kontext basierend auf der betrieblichen Kategorisierungstechnik in 4 verschiedene Kategorien:

1. Messbar: Informationen die aus direkt messbaren Werten bestehen. Diese ändern sich oft oder gar kontinuierlich.
2. Statisch: Informationen die sich während der Lebenszeit eines Systems gleich bleiben.
3. Profiliert: Informationen die sich selten ändern.

4. Abgeleitet: Informationen die unter Verwendung anderer Daten gewonnen wurden.

Van Bunningen et al. [17] ordnen Kategorisierungsversuche in zwei übergeordnete Gruppen: Betrieblich und Konzeptionell.

1. Betriebliche Kategorisierung: Einordnung anhand dessen wie der Kontext akquiriert, modelliert und behandelt wird.
2. Konzeptionelle Kategorisierung: Einordnung anhand der Bedeutung des Kontextes und der konzeptionellen Beziehungen

Chong et al.[6] schlägt Historie als Kontextkategorie vor. Dabei werden der zeitliche Verlauf, der Werte die eine bestimmte Messgröße in der Vergangenheit angenommen hat, als Kontext definiert. Das erlaubt die Festlegung von Standardwerten. Damit ist unter Umständen eine Vorhersage darüber welche Werte die Messgrößen zukünftig annehmen werden möglich.

# 3 Voraussetzungen und verwandte Arbeiten

**Situation** Eine Situation

**Umgebung** Umgebung ist ein Synonym für Kontext [2].

**Policy**

**Netzwerk-Sicherheits-Monitoring**

**Intrusion Detection** Angriffserkennung ist "der Prozess der Überwachung von Ereignissen in einem Computersystem oder Netzwerk, und die Analyse dieser Ereignisse auf Anzeichen eines möglichen Zwischenfalls. Gemeint sind damit Verstöße oder unmittelbare Bedrohungen von Sicherheitsrichtlinien, Akzeptanzrichtlinien oder Standardsicherheitspraktiken"[15].

**Intrusion Detection System** Auf Grundlage der Definition von Intrusion Detection, ist ein IDS somit Software die den Prozess, einen Eingriff zu erkennen, automatisiert [15].

## 4 Design

Zuerst wird die Notwendigkeit für eine Taxonomie erläutert. Dann erfolgt die Definition der Taxonomiekategorien. Danach wird festgelegt in welcher Form die Kontextinformationen in den einzelnen Kategorien vorliegen müssen und wie man Informationen aus unterschiedlichen Quellen in einer für sammelt. Zusätzlich wird noch darauf eingegangen welche Anforderungen ein IDS erfüllen sollte, welche davon im speziellen für diese Arbeit relevant sind und welche Schwächen des IDS mithilfe des vorgeschlagenen Designs und Kontextsensitivität gelöst werden.

### 4.1 Notwendigkeit einer Taxonomie

Die Evaluation verschiedener Kategorisierungsschemata zeigt das keine Kategorisierung allen Ansprüchen gerecht werden kann [14].

### 4.2 Erstellung einer Taxonomie

Es folgt eine Kombination bereits vorgeschlagener Kategorisierungsschemata. Dazu lege ich zuerst die Definition der einzelnen Kategorien im Bezug auf kontextsensitive Zugriffskontrolle fest. Die Kategorien müssen dafür, abhängig davon mit welchem Fokus sie der jeweilige Autor konstruiert hat, mehr oder weniger stark angepasst werden. Die Grafiken 4.1 und 4.2 veranschaulichen die Beziehung der einzelnen Kategorien zueinander und geben Beispiele dafür wie Kontextinformationen aussehen können. Als Vorbild diente [14].

#### 4.2.1 Mensch

Die offensichtliche Kategorisierung von Kontext, und auch von den meisten Autoren auf die im Kapitel 2 erwähnt wurden, ist die aus der Perspektive des Nutzers also einer Person. Auch für eine Einordnung im Rahmen der Zugriffskontrolle lässt sich argumentieren das ein Fokus auf Menschen sinnvoll ist. Zwar Rücken gerade in diesem Bereich Personen gelegentlich in den Hintergrund und der überwiegende Teil des Netzwerkverkehrs wird ausgelöst ohne das Nutzende davon etwas mitbekommt. Aber in letzter Instanz sind sowohl Verursachende, Überwachende und Forschende Menschen. Menschen haben vielfältige Anliegen, müssen unauffälligen von auffälligem Netzwerkverkehr unterscheiden, versuchen die Beweggründe anderer indirekt zu erschließen, zu kategorisieren und in Taxonomien zu verpacken.

## Konzeptionell

Einordnung anhand der Bedeutung des Kontextes und der begrifflichen Beziehungen.

**Primär** Kontextinformationen die gesammelt werden können ohne bereits vorhandene Daten zu verwenden oder zu kombinieren [2].

**Sekundär** Kontext der durch das verarbeiten von primärem Kontext erschlossen werden kann. Dies kann durch die Kombination einzelner Datenpunkte einer oder mehrerer Kategorien oder durch Abfragen weiterer Informationen mithilfe der primären Informationen geschehen [2].

## Betrieblich

Wie schon in der Analyse bereits vorgeschlagener Kategorien bedeutet die betriebliche Kategorisierung: "Einordnung anhand dessen wie der Kontext akquiriert, modelliert und behandelt wird "[17].

**Zeit** Zu welcher Zeit ein Zugriffsanfrage erfolgt.

**Ort** Von welchem Ort eine Zugriffsanfrage stammt.

**Identität** Wer Zugriff auf eine Ressource erfragt.

**Aktivität** Was in einer Situation passiert bzw. welche Aktion eine Entität ausführt.

**Grund** Warum etwas getan wird. In Abbildung 4.1 ist lediglich eine sekundärer Grund mit einem Beispiel versehen da ein Grund für etwas ausschließlich aus anderen Informationen erschlossen werden kann und nicht im Netzwerk vorliegt.



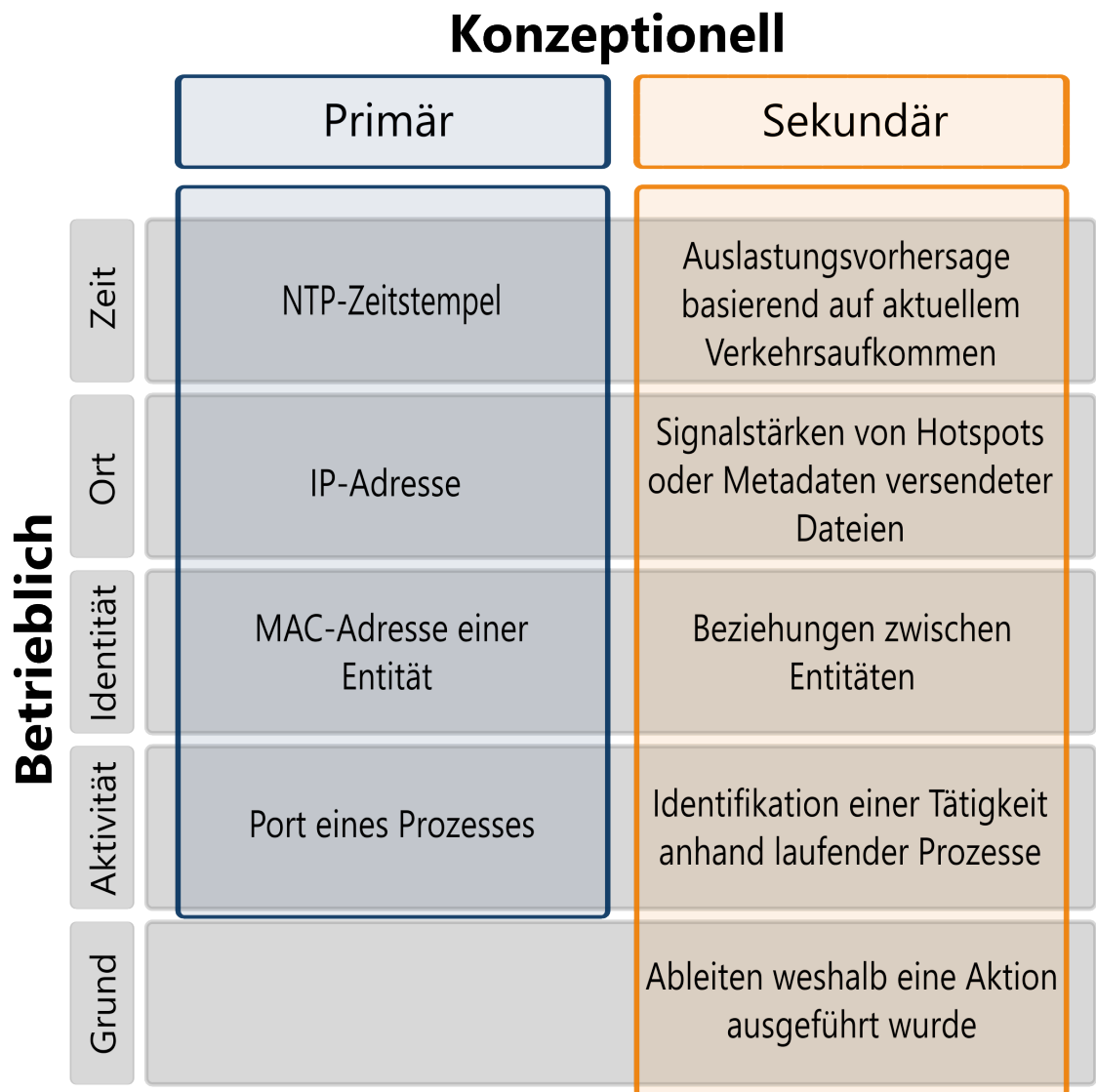


Abbildung 4.1: Nutzerzentrierte Kategorisierung

#### 4.2.2 Netzwerk

Ein Netzwerk besteht im allgemeinen aus:

1. Entitäten die daran teilnehmen
2. Kommunikation zwischen den Entitäten
3. Annahmen bzw. Normen bezüglich:
  - a) des Verhaltens und der Eigenschaften der Entitäten
  - b) der Form und dem Inhalt der Kommunikation

In Rahmen der Zugriffskontrolle erfolgt Kommunikation im Netzwerk über spezifische Protokolle zwischen verschiedenen Entitäten die durch bestimmte Attribute charakterisiert werden. Die Normen werden dabei initial festgelegt und im Verlauf der Lebenszeit des Netzwerkes angepasst.

## Protokolle

Kategorisierung anhand des verwendeten Kommunikationsprotokolls. Orientiert sich am ISO/OSI-Referenzmodell [7]. Die Zuordnung zu einer bestimmten Schicht und damit Kategorie erfolgt anhand der für die einzelnen Schichten üblichen Protokolle. Das ermöglicht die Identifikation von Entitäten anhand der von ihnen genutzten Protokolle. So kann beispielsweise ein Switch oder Router von einem Nutzer oder einer Anwendung unterschieden werden.

**Anwendung** Beinhaltet allen Netzwerkverkehr der sich der Sitzungsschicht, Darstellungsschicht oder Anwendungsschicht zuordnen lässt

**Transport** Pakete die sich der Transportschicht zuordnen lassen.

**Vermittlung** Netzwerkverkehr der zur Vermittlungsschicht gehört.

## Entitäten

Kategorisierung von Kontextinformationen abhängig davon welche Art von Entität sie betreffen. Diese Unterscheidung setzt genauso wie die Historie eindeutig identifizierbare Entitäten voraus.

**Gerät** Informationen die sich auf ein spezifisches Gerät beziehen

1. Ein Gerät ist beispielsweise einen Router oder das Endgerät eines Nutzers.
2. Informationen können beispielsweise die Liste an installierter Software, die Menge laufender Prozesse oder die Auslastung der Hardwarekomponenten sein.

**Nutzer** Informationen die sich auf einen Nutzer beziehen.

1. Ein Nutzer im Sinne eines Computersystems und keine physische Person. So kann eine physische Person durchaus mehrere verschiedene logische Nutzerkonten haben.
2. Informationen können zum Beispiel die Zugriffsrechte eines Nutzers sein.

**Anwendung** Informationen die sich auf eine Anwendung beziehen.

1. Anwendung umfasst jegliche Softwareprozesse die für sich oder in Kombination einen bestimmten Zweck erfüllen.
2. Eine Anwendung erzeugt für sie charakteristischen Netzwerkverkehr, versendet oder empfängt also bestimmte Informationen.

## Policies

Einordnung der Kontextinformationen abhängig davon ob sie der für den Anwendungsfall definierten Normen entsprechen.

**Erwartet** Form der Kommunikation mit verschiedenen Protokollen oder Verhalten von Entitäten entsprechend den im Netzwerk geltenden Vorschriften.

**Ungewöhnlich** Form der Kommunikation mit verschiedenen Protokollen oder Verhalten von Entitäten die in Kombination mit den im Netzwerk vorherrschenden Bedingungen entweder auffällig oder irrational sind.

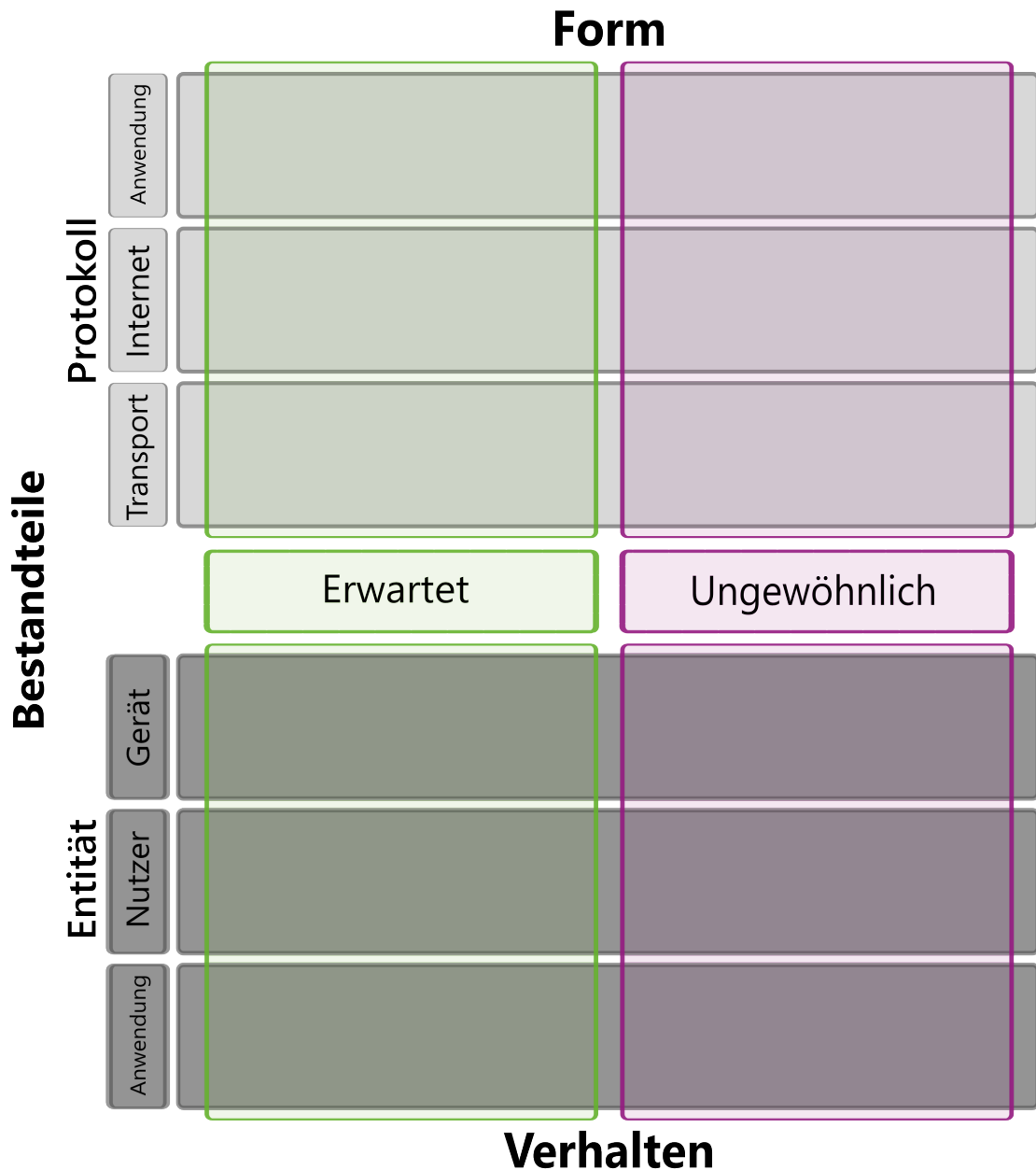


Abbildung 4.2: Kategorisierung von Kontextinformationen als Netzwerkbestandteile

### 4.2.3 Historie

Unabhängig vom gewählten Fokus bzw. Blickwinkel auf die Kategorisierung von Kontext benötigt man, um bereits gesammelten Kontext nutzen zu können, eine Art Gedächtnis. Im Fall eines IDS welches ohnehin, aufgrund seiner Funktionsweise, über einen längeren Zeitraum Informationen verarbeitet und neue Erkenntnisse speichert, bietet sich eine Historie wie in Kapitel 2 beschrieben an. Die Historie einer Entität wird in dieser Taxonomie zweigeteilt. Sie besteht aus:

1. Einem aktiven Teil also ihrem Verhalten, beispielsweise früheren Verbindungen bzw. Verbindungsanfragen
2. Einem passiven Teil also dem Zustand der für sie charakteristischen Attribute, beispielsweise einem Nutzernamen oder die Versionsnummer eines bestimmten Programms

## Dynamik

Es liegt in der Natur der Sache das sich die Werte, aus denen sich die Historie einer Entität zusammensetzt, je nachdem welchem Teil sie zugeordnet werden, verschieden oft ändern. Statische Messgrößen ändern dabei ihre Werte nie oder nur sehr selten. Dynamische Messgrößen hingegen sehr oft. In diesem Fall wird eine Unterteilung in jährlich, monatlich, wöchentlich, täglich, stündlich, minütlich und sekundlich vorgenommen.

## Raten

Die Historie ist weiterhin in 3 verschiedene Bereiche unterteilt. Abhängig davon wo die Änderung auftritt und ob das IDS oder eine Entität die Aktualisierung des Wertes auslöst.

**Änderungsrate** Gibt an wie oft eine Entität eine Werteänderungen mitteilt.

**Abtastrate** Wie oft Werte einer Entität vom IDS abgefragt bzw. aktualisiert werden.

**Abfragerate** Wie oft Werte im Netzwerk von einer Entität, die nicht das IDS ist, abgefragt werden.

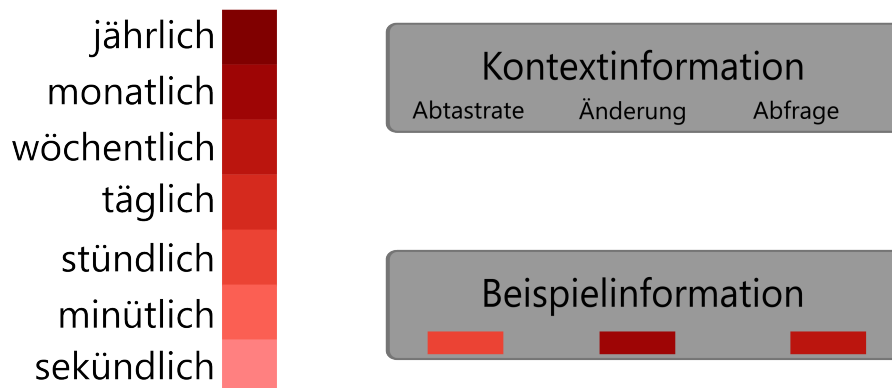


Abbildung 4.3: Farblegende der Dynamik und Historie eines Beispieleintrages

## 4.3 Form des Kontextes

Nachdem die einzelnen Kategorien definiert wurden muss festgelegt werden in welcher Form die Kontextinformationen vorliegen sollen bzw. gebracht werden müssen. Für fast alle Kategorien selbsterklärend. Protokollkontext hat eine durch das Protokoll selbst vorgegebene feste Form, ein Zeitstempel ebenso. Kurz gesagt: Größtenteils ist relativ offensichtlich und teils sehr sehr spezifisch welche Form die Information haben sollte. Deshalb wird an dieser Stelle nicht näher darauf eingegangen.

Lediglich die Historie bildet eine Ausnahme. Hier gibt es je nach Bedarf des Anwendungsfalls einen gewissen Spielraum. Die gespeicherten Informationen werden abhängig von der Implementierung eventuell gar nicht vom IDS eingelesen oder gar selbst vom IDS erzeugt. Auch gibt es höchst unterschiedliche Ansprüche an Form, Umfang und Zeitraum oder Beschränkungen hinsichtlich Performanz und Speicherbedarf.

### 4.3.1 Historie

Die Historie muss genug Informationen enthalten, um damit neue Entscheidungen in der Gegenwart oder Zukunft zu treffen und sie eindeutig zu identifizieren können. Schließlich kann eine Historie die dabei helfen soll Entscheidungen zu treffen die den Netzwerkverkehr von Entitäten betreffen kann nicht ohne eindeutig identifizierte Entitäten funktionieren.

Die Identifikation erfolgt anhand der Headerinformationen der einzelnen Kommunikationsschichten. In der Historie gespeichert werden entweder die gesamte Payload eines Pakets oder zumindest ein ausreichend aussagekräftiges Subset. Die Regelmäßigkeit mit der die Historie aktualisiert bzw. erweitert wird ist vom jeweiligen Gegebenheiten und Anforderungen abhängig. Nach Ansicht des Autors sollte ein Subset mindestens folgende Daten beinhalten, um eine ausreichende Rekonstruktion des Kenntnisstandes, mit dem die Zugriffsentscheidungen ursprünglich getroffen wurden, zu ermöglichen und damit aussagekräftig zu sein:

1. Zur Identifikation einer Entität und der Zuordnung ihrer Verbindung:
  - a) MAC-Adresse
  - b) IP-Adresse
  - c) Port
2. Zur Rekonstruktion der ursprünglichen Zugriffsentscheidungen die die Entität betreffen:
  - a) MAC-Adresse des Ziels
  - b) IP-Adresse des Ziels, gesetzte Flags, TTL, Protokoll
  - c) Port des Ziels, protokollspezifische Informationen

## 4.4 Kontextgewinnung

Die für die in 4.2 enthaltenen Kategorien gewinnt man benötigte Informationen entweder direkt aus dem Netzwerkverkehr oder Dabei sollte man bedenken an welcher Stelle und wie oft man Informationen abrufen oder automatisch aktualisiert. [14]

Es folgt eine Übersicht einiger Beispiele für Möglichkeiten Kontextinformationen zu gewinnen. - osquery - network (open ports) | Nessus - devices in network | Configuration Management Database (CMDB), nmap - cve reference - yes/no - protocol header

## 4.5 Umwandlung der Taxonomie in IDS-Signaturen

Um die im Abschnitt Kontexttaxonomie festgelegten Kategorien in für ein IDS nutzbare Form zu bringen gilt es gewisse Dinge zu beachten:

Die Kontextsensitivität eines Computers unterscheidet sich drastisch von der eines Menschen. Rechensysteme sind sehr gut darin Daten zu erfassen und zu sammeln, aber Menschen sind immer noch nötig um verschiedene Kontexte zu erkennen und zu entscheiden welches Handeln in einer bestimmten Situation angemessen ist [8]. Der limitierende Faktor

Tabelle 4.1: Placeholder

Eigenschaft	IDS Typ	Beschreibung
Plattform	Host	Überwacht Aktivitäten auf dem System auf dem es eingesetzt wird um lokale Angriffe zu erkennen.
	Netzwerk	Überwacht Aktivitäten im Netzwerk um Angriffe, die über eine Netzwerkverbindung ausgeführt werden zu erkennen.
	Hybrid	Kombiniert host- und netzwerk-basierte Intrusion Detection Systeme.
Angriffserkennung	Signatur	Überwacht System- und/oder Netzwerkaktivitäten anhand einer Reihe von Signaturen bekannter Angriffe auswertet. Daher ist es nicht in der Lage, Zero-Day-Angriffe zu erkennen, d. h. Angriffe, die Schwachstellen ausnutzen, die vor der Ausführung der Angriffe nicht öffentlich bekannt sind.
	Anomalie	Verwendet ein Basisprofil regulärer Netz- und/oder Systemaktivitäten als Referenz, um zwischen regulären und auffälligen Aktivitäten zu unterscheiden, wobei letztere als Angriffe behandelt werden. Wird typischerweise durch die Überwachung regulärer Aktivitäten trainiert, um ein Basisaktivitätsprofil zu erstellen.
	Hybrid	Verwendet sowohl signatur-basierte als auch anomalie-basierte Angriffserkennungsmethoden.
Struktur	Zentral	Kann nur an einem einzigen Standort eingesetzt werden.
	Verteilt	Besteht aus mehreren Teilsystemen für die an verschiedenen Standorten eingesetzt werden können und miteinander kommunizieren, um für die Erkennung von Angriffen relevante Daten, z. B. Angriffswarnungen, auszutauschen. Kann koordinierte Angriffe auf mehrere Standorte in einer bestimmten zeitlichen Abfolge erkennen.

des Potenzials eines kontextsensitiven Systems ist das Maß an Kontext das ein Entwickler vorhersehen und kodieren kann. Es ist aber weder beim Design noch später bei der Implementierung unmöglich alle Zusammenhänge vorherzusehen. Dementsprechend schwer wird ist es ein in sich geschlossenes und allumfassendes Regelset festzulegen [14]. Nach Greenberg et al. [9] gibt drei non-triviale Hauptaspekte die man beim Entwerfen eines kontextsensitiven Systems beachten sollte:

1. Spezifizieren aller möglichen Kontextzustände
2. Wissen welche Informationen einen konkreten Kontextzustand akkurat festlegen.
3. Welche Aktion im jeweiligen Zustand ausgeführt werden sollen.

#### 4.5.1 Entscheidung für ein IDS

IDSs können anhand der überwachten Plattform, der verwendeten Erkennungsmethode und der Struktur in der sie eingesetzt werden kategorisiert werden. [13].

### 4.5.2 Verbesserung der Schwächen

Der Hauptnachteil eines signatur-basierten IDS ausschließlich Angriffe zu erkennen die vordefinierten Verhaltensmustern entsprechen. Der Hauptnachteil eines anomalie-basierten IDS ist die Notwendigkeit die Baseline, mitunter sehr oft zu aktualisieren. Um diese Nachteile auszugleichen bietet es sich an die bereits vorhandenen, gesammelten und aufbereiteten Kontextinformationen zu verwenden. Dies ermöglicht die schon vorhanden Signaturen so zu verbessern das sie eine größere Menge von Ereignissen abdecken oder bereits definierte Ereignisse genauer zu spezifizieren und so weniger (falsche) Meldungen zu generieren ohne dabei das Verhalten des Netzwerkverkehrs dauerhaft neu bewerten zu müssen.

## 4.6 Anspruch an das IDS

Um die Performance des konkret gewählten IDS verbessern zu können, muss man festlegen welche Kriterien die Leistung beeinflussen bzw. bestimmen und wie man diese gewichtet. Mell et al.[12] nennen in ihrer Übersicht verschiedene Charakteristiken zur quantitativen Bestimmung der Erkennungsgenauigkeit eines IDS und erläutern zusätzlich die Wechselwirkungen zwischen einzelnen Kriterien, die beim Vergleich verschiedener IDS-Lösungen beachtet werden sollten.

### Abdeckung

Gibt an welche Typen von Angriffen ein IDS unter idealen Bedingungen erkennen kann.

### Wahrscheinlichkeit falscher Alarme

Gibt die Wahrscheinlichkeit das durch ein IDS ausgelöste Alarme durch gutartigen bzw. nicht-schädlichen Netzwerkverkehr verursacht wurden an.

$$\text{Rate an falsch – Positiven Meldungen} = \frac{\text{Anzahl falscher Alarme}}{\text{Anzahl aller Alarme}}$$

### Wahrscheinlichkeit einer Erkennung

Gibt die Rate der durch das IDS korrekt erkannten Angriffe an.

$$\text{Erkennungswahrscheinlichkeit} = \frac{\text{Anzahl korrekt erkannter Angriffe}}{\text{Anzahl aller Angriffe}}$$

### Resistenz

Ein signatur-basiertes IDS bzw. der menschliche Administrator hinter dem System weisen Probleme auf die nicht direkt beim Umgang mit verarbeitetem Netzwerkverkehr, sondern schon bei der bewussten, unbewussten, oder erzwungenen Entscheidung welcher Netzwerkverkehr überhaupt in Frage kommt, entstehen:

1. Ein zu große Menge an zu verarbeitendem Netzwerkverkehr die die Verarbeitungskapazität eines IDS übersteigt kann dazu führen das Netzwerkpakete verworfen und Angriffe nicht erkannt werden
2. Pakete die zwar nicht bösartig sind aber so konstruiert das sie möglichst viele IDS Signaturen auslösen, überfordern den Administrator oder stören eventuell sogar die Verarbeitung von Paketen generell.
3. Ein Angreifer könnte eine Vielzahl "harmloserer" aber trotzdem noch als schädlich zu deklarierende Pakete senden um einen größeren Angriff im Netzwerkverkehr zu verschleiern.
4. Pakete die möglicherweise vorhandene Fehler im IDS selbst ausnutzen.



### **Korrelation zwischen Einzelereignissen**

Demonstriert wie gut ein IDS eine Korrelation zwischen einzelnen Ereignissen, möglicherweise verschiedenen Ursprungs herzustellen. Die Ereignisse können dabei aus Routern, Firewalls, Anwendungen, dem IDS selbst oder einer großen Bandbreite anderer Quellen stammen.

### **Unbekannte Angriffe vorhersehen**

Gibt an wie gut ein IDS einen Angriff erkennt der so noch nicht aufgetreten ist. Signaturbasierte IDS sind allgemein, mit wenigen Ausnahmen, nicht in der Lage solch einen Angriff zu erkennen. Normalerweise erhöht die Fähigkeit eines Systems einen noch unbekannten Angriff zu erkennen, im Vergleich zu Systemen die dies nicht versuchen, zusätzlich die Rate an falsch-positiven Meldungen.

### **Identifizieren von Angriffen**

Wie gut ein IDS einem Angriff den es erkennt, einen Namen oder eine Kategorie, beispielsweise ein CVE-Nummer, zuordnen kann.

### **Beurteilung eines Angriffs**

Indikator dafür ob ein IDS den Erfolg und die Auswirkungen eines Angriffes korrekt beurteilen kann. In aktuellen Netzwerkumgebungen schlagen viele Angriffs(-versuche) fehl. Die meisten IDS unterscheiden allerdings nicht zwischen erfolgreichen und fehlgeschlagenen Angriffen. Für den selben Angriff können manche IDS die Anzeichen dafür ob ein Angriff erfolgreich war erkennen, andere lediglich das ein Angriff stattgefunden hat, allerdings ohne feststellen zu können ob er erfolgreich war. Die Fähigkeit, den Grad zu dem ein Angriff auf das überwachte System erfolgreich war zu beurteilen ist essenziell. Eine Vorfilterung der Meldungen durch das IDS vereinfacht die Arbeit des Netzwerkadministrators bzw. Analysten stark, da so eine Analyse des Angriffsszenarios und der Korrelation einzelner Angriffe vereinfacht wird. Diese Fähigkeit bei einem gegebenen IDS messen zu können setzt das Wissen, darüber welche Angriffe erfolgreich sind und welche nicht, voraus.

## **4.6.1 Einordnung der Kriterien**

Die Wahrscheinlichkeit einen Angriff zu erkennen variiert mit der Rate an falsch-positiven Meldungen. Bei der Konfiguration eines IDS kann entweder die Falsch-positiv-Rate oder die Erkennungsrate optimiert werden.

## **4.6.2 Auswahl der Kriterien**

Diese Arbeit beschäftigt sich ausschließlich mit sicherheitsrelevanten Leistungsmetriken. Dabei soll hauptsächlich der Einfluss von Kontextsensitivität auf die Abdeckung, die Erkennungswahrscheinlichkeit und die Rate falsch-positiver Alarme betrachtet werden. Zusätzlich wird beleuchtet ob zusätzlicher Kontext die Interpretierbarkeit von Meldungen durch einen menschlichen Nutzer und das Identifizieren von Angriffen verbessert.

# 5 Implementierung

Der im Kapitel 4 dargelegte Aufbau eines Netzwerkes findet sich auch in der Implementation wieder. Scapy, Wireshark und Zeek verarbeiten Kommunikationsdaten, zeek-agent stellt die Attribute der Entitäten bereit und der Skript-schreibende legt die geltenden Normen durch die erstellten Skripte und das dadurch abgedeckte Verhalten fest.

## 5.1 Erläuterung der wichtigsten Komponenten

Eine kurze Vorstellung der wichtigsten Komponenten der Implementierung insofern sie im Rahmen der einzelnen Schritte des Versuchsaufbaus essenziell sind.

### 5.1.1 Zeek

Ein passives, quelloffenes Analysewerkzeug für Netzwerkverkehr das via eigener Skriptsprache unter anderem folgendes ermöglicht:

1. Implementierung beliebiger Analyseaufgaben
2. Interpretation von Netzwerkverkehr
3. Erstellung von Protokollen auf der Grundlage dieses Verkehrs

Zeek ist dabei weder rein signatur-basiert wie Suricata noch ausschließlich als Protokollanalytiker im Sinne von Wireshark zu verstehen. Vielmehr handelt es sich bei Zeek um eine Mischung die eine kompakte, aber dennoch detailgetreue Darstellung von Netzwerkprotokollen ermöglicht. Dies erlaubt ein besseres Verständnis des Netzwerkverkehrs und der Netzwerknutzung [1].

### 5.1.2 Zeek-Agent

Zeek-Agent ist ein Endpunkt-Agent der Informationen, für zentrales Monitoring an Zeek sendet. Abfragen kann man verschiedene Aktivitäten eines Hostsystems, darunter zum Beispiel aktuell laufende Prozesse, offene Sockets oder den Inhalt bestimmter Dateien. Diese erscheinen in Zeek, genauso wie Netzwerkaktivität, als Ereignisse und können so in Skripten verwendet werden [20].

## 5.2 Versuchsaufbau

Der Ablauf für alle Anwendungsfälle ist grundsätzlich sehr ähnlich:

1. Erzeugung
2. Mitschnitt
3. Analyse
  - a) Einlesen von Netzwerkverkehr
  - b) Einbindung des zusätzlichen Kontextes
  - c) Logging

### 5.2.1 Erzeugung

Der Netzwerkverkehr wurde mittels Scapy erzeugt. Das ermöglicht den für die verschiedenen Szenarien benötigten Netzwerkverkehr zu erzeugen und die einzelnen Schichten eines Pakets an den jeweiligen Anwendungsfall anzupassen. In ist dieser Prozess ausschnittsweise dargestellt. Eine Übersicht über alle dafür verwendeten Skripte findet sich im Anhang.

Quelltext 5.1: Konfiguration und Versendung eines Pakets

```

6 def send_packet(ip_address_src, ip_address_dst):
7     source_server = ip_address_src
8     target_server = ip_address_dst
9     layer_2 = Ether()
10    layer_3 = IP(src=source_server, dst=target_server)
11    layer_4 = TCP(sport=80, dport=43468)
12    tcp_pkt = layer_2 / layer_3 / layer_4
13    sendp(tcp_pkt)

```

### 5.2.2 Mitschnitt

Der Mitschnitt der von Scapy versendeten Pakets erfolgt mit Wireshark. Die mitgeschnittenen Pakets werden im als pcap-Dateien gespeichert um das einlesen in Zeek zu ermöglichen.

### 5.2.3 Logging

In Zeek erfolgt das Schreiben in Logs immer nach dem selben Prinzip:

1. Vor dem Ausführen eines Skriptes wird ein Log und die darin zu speichernden Informationen festgelegt (Z. 3-10)
2. Nutzer-definierter Log wird initialisiert (Z. 19)
3. Log-Eintrag für den Log wird definiert (Z. 20)
4. Log-Eintrag wird der Verbindung als Information hinzugefügt (Z. 23)
5. Eintrag wird in den Log geschrieben (Z. 24)

Quelltext 5.2: Generierung einer Log-Datei mit Verbindungsinformationen

```

1 export {
2   # Create an ID for our new stream. By convention, this is called "LOG".
3   redef enum Log::ID += { LOG };
4
5   # Define the record type that will contain the data to log.
6   type Info: record {
7     timestamp: time &log;
8     id: connection_id &log;
9     notice: string &log;
10  };
11 }
12
13 redef record connection += {
14   # By convention, the name of this new field is the lowercase name
15   # of the module.
16   examplelog: Info &optional;
17 };
18 event zeek_init(){
19   Log::create_stream(ExampleModule::LOG, $columns=Info, $path="examplemodule");
20   local record: ExampleLog::Info = $ts=current_time(), $id=c$id,
21     $notice="Example Notice"];
22   # Store a copy of the data in the connection record so other
23   # event handlers can access it.
24   c$examplelog = record;
25   Log::write(ExampleModule::LOG, rec);
26 }

```

## 5.3 Skripte

Der Kern der Implementierung sind Zeek Skripte. Hier werden die gesammelten Kontextinformationen verwendet. Nachfolgend werden für einige ausgewählte Kategorien Anwendungsfälle vorgestellt.

### 5.3.1 Geografische Koordinaten und Ortszeit

Das Volumen von durch Menschen verursachten Netzwerkverkehr ist meist tageszeitabhängig. So wird in der Regel nachts weniger kommuniziert als am Tag. Deshalb erscheint es sinnvoll den Grenzwert für erzeugtes Verkehrsaufkommen, ab dem ein Sender als potenziell böswillig eingestuft wird, je nach Uhrzeit anzupassen.

Zeile 32 Zuordnung einer IP-Adresse zu einem geographischen Ort.

Zeile 36 - 42 Ermittlung der Uhrzeit am Ursprung der Anfrage mithilfe der lokalen Uhrzeit in Kombination mit dem aus dem Abstand ermittelten Zeitunterschied.

Zeile 45 Anpassung des Grenzwertes

Quelltext 5.3: Geolokalisierung und Setzen des Grenzwertes

```

31 function geolocation(c: connection):double{
32   local origin_longitude = lookup_location(c$id$orig_h)$longitude;
33   return origin_longitude;
34 }
35
36 function time_at_geolocation(longitude: double): int{

```

```

37     local time_to_add = (longitude - home_longitude)*240;
42     return useable_time_at_origin;
43 }
44
45 function set_threshold(c_time: int): double{
46     if(c_time< opening_time || c_time > closing_time )
47         threshold = threshold-night_time_decrease;
48     else
49         threshold = threshold+day_time_increase;
50     return threshold;
51 }

```

### 5.3.2 Verwendete Ports

Wenn eine Verbindung oder der Versuch eine Verbindung zu etablieren, mit einem bestimmten Port des Systems als Ziel beobachtet wird ohne das ein Prozess gibt der diesem Port mittels zugeordnet werden kann, wird die Verbindung bzw. der Verbindungsversuch im dazugehörigen Log vermerkt.

1. Das Skript erfagt bei den zeek-agents in regelmäßigen Abständen die auf Hostsystemen laufenden Prozesse und deren verwendete Ports
2. Das Skript vergleicht den Zielport jeder eingehenden Verbindung mit der Liste von Hostports
3. Abhängig vom Ergebnis wird eine Meldung in den Log geschrieben

Quelltext 5.4: Abfrage und Abgleich der Ports

```

52 function check_outgoing_connection(c:connection){
53     local _port = port_to_count(c$id$orig_p);
54     if(_port !in local_ports){
55         local rec: Querytest::Info = [$ts=current_time(), $id=c$id, $notice="No
           Application running on this port"];
63 event users_result(ctx: ZeekAgent::Context, data: Columns){
64     local new_entry : count;
65     local connection_port = data$remote_port;
66     new_entry = connection_port;
67     local_ports[new_entry] = data$name;
68 }
77
       local test_query_join = ZeekAgent::query([$sql_stmt=str_stmt_join,
           $event=query_event, $schedule=_schedule]);

```

### 5.3.3 DNS-Auflösung

Menschliche Nutzer verwenden, bei der Nutzung ihres Endgerätes, im Gegensatz zu Computern, keine IP-Adressen um im Suchanfragen zu formulieren. DNS ordnet IP-Adressen menschenfreundliche Domainnamen zu. Wenn sich das Gerät eines Nutzers mit einem anderen verbindet geht dem eine DNS-Anfrage von diesem Gerät an einen DNS-Nameserver voraus oder die IP-Adresse ist in lokalen Konfigurations-Dateien vermerkt. Wenn beispielsweise eine TCP-Verbindung eines Webbrowsers beobachtet wird, ohne das eine zuordenbare DNS-Anfrage erfolgt ist oder die IP-Adresse in auf dem Endgerätes vermerkt ist, ist das

in den meisten Fällen ein Grund zum Handeln. Das Skript erfragt periodisch die hosts-Datei eines Unix-Systems (Z. 63).

Loggt für jede Antwort eines DNS-Servers die aufgelöste URL und dazugehörige IP-Adresse (Z. 56).

Im Log vermerkt sind ausgehende Verbindungen deren Zieladressen vorher nicht durch einen DNS-Server oder die hosts-Datei aufgelöst wurden (Z. 68).

#### Quelltext 5.5: Überprüfung der Verbindungsziele eines Endgerätes

```

53 event query_result(ctx: ZeekAgent::Context, data: Columns){
54     local ip_address = to_addr(data$line_content[0]);
55     local host_name = data$line_content[1];
56     resolved_addresses[ip_address] = host_name + " from local hosts";
57 }
58
59 function query_hosts_file(){
60     local str_stmt_hosts = "SELECT columns FROM
61         files_columns(\"/etc/hosts\", \"$1:text,$2:text\");
62     local query_event = query_result;
63     local _schedule = 30 secs;
64     local test_query_join = ZeekAgent::query([$sql_stmt=str_stmt_hosts,
65         $event=query_event, $schedule=_schedule]);
66 }
67
68 event check_resolve_table(c : connection){
69     local destination_ip = c$cid$resp_h;
70     if(destination_ip !in resolved_addresses && destination_ip !in dns_server){
71         local rec: DNStest::Info = [$ts=current_time(), $id=c$cid,
72             $notice="Connection without Resolve!"];
73         c$dnstest = rec;
74         Log::write(DNStest::LOG, rec);
75     }
76 }

```

## 6 Evaluation

Zuerst einmal sollte im Rahmen dieser Arbeit geklärt werden, ob und wie Kontext kategorisiert werden kann. Untersucht wurde ebenfalls welche Kontextinformationen, zur Anpassung der Arbeitsabläufe eines IDS, verwendet werden können. Dabei wird zusätzlich beurteilt

Die erhöhte Interpretierbarkeit wird durch die Kombina

Zeek bietet die beste Software/Umgebung für die Entwicklung neuer Erkennungs- oder Verarbeitungstechniken. Es kann für die kontinuierliche Überwachung von Netzwerken mit hohem Durchsatz verwendet werden. Die Skripting-Umgebung ist in einer speichersicheren Sprache erweiterbar, die auf die Verarbeitung von Netzwerkdaten spezialisiert ist. Im Gegensatz zu anderen Tools, ist es nicht auf ein einziges Paradigma für die Netzüberwachung limitiert.

### 6.1 Mächtigkeit eines kontextsensitiven IDS

### 6.2 Vergleich der Kontextkategorien

#### 6.2.1 Verfügbarkeit der Informationen

Die Informationen die ich in den definierten Kategorien als gegeben vorausgesetzt habe sind in der Praxis unterschiedlich schwer zu akquirieren. Offensichtlich ist es um ein Vielfaches einfacher korrekt die aktuelle Uhrzeit zu ermitteln als den Grund dafür das ein Nutzer eine Aktion ausführt zu bestimmen. Die nachfolgende Grafik dient dazu den meiner Einschätzung nach benötigten Aufwand zur Beschaffung einer Information zu veranschaulichen.

		<b>Konzeptionell</b>	
		Primär	Sekundär
<b>Betrieblich</b>	Zeit	NTP-Zeitstempel	Auslastungsvorhersage basierend auf aktuellem Verkehrsaufkommen
	Ort	IP-Adresse	Signalstärken von Hotspots oder Metadaten versendeter Dateien
	Identität	MAC-Adresse einer Entität	Beziehungen zwischen Entitäten
	Aktivität	Port eines Prozesses	Identifikation einer Tätigkeit anhand laufender Prozesse
	Grund		Ableiten weshalb eine Aktion ausgeführt wurde

Abbildung 6.1: Placeholder



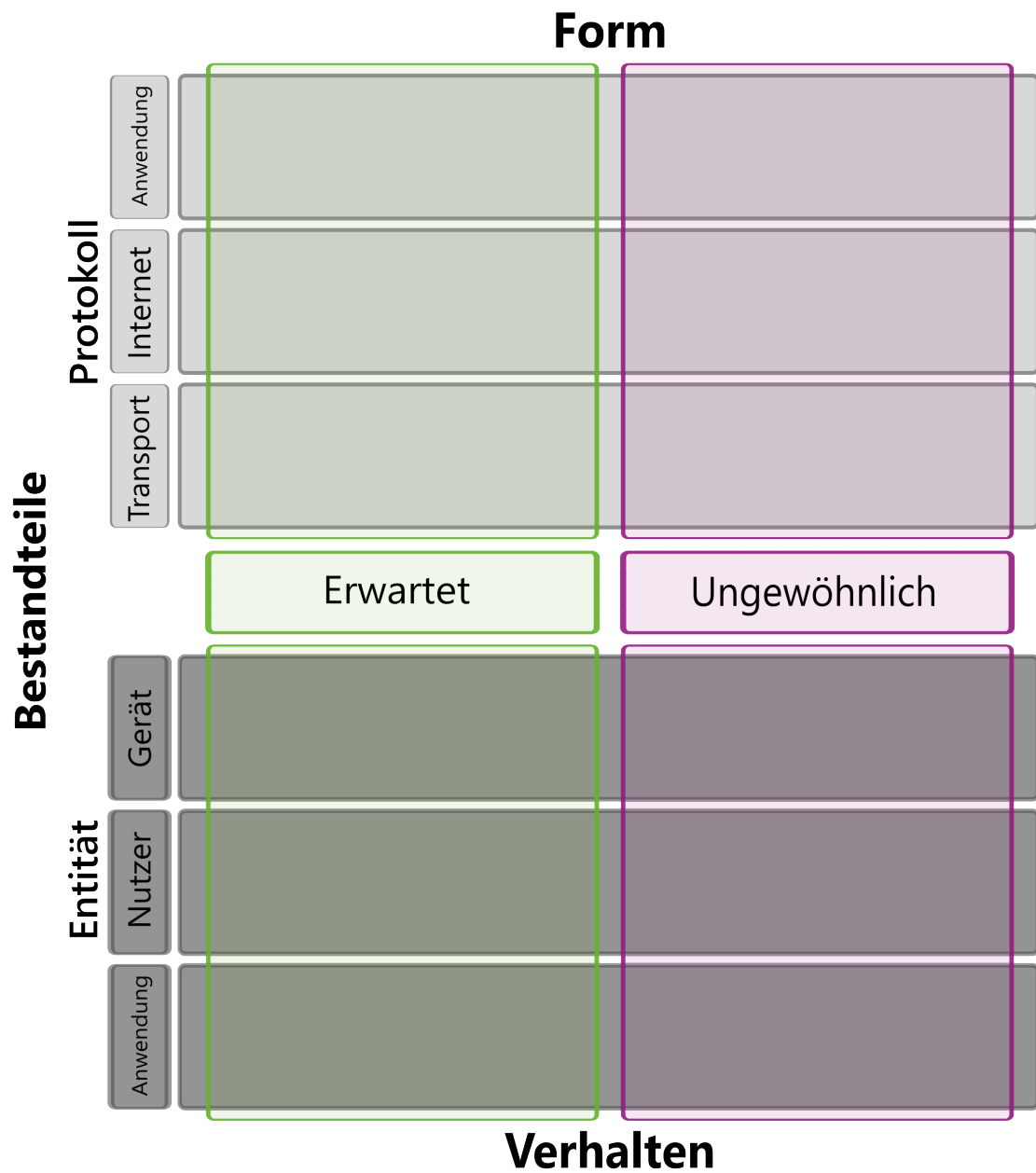


Abbildung 6.2: Placeholder

### 6.2.2 Qualität der Informationen

Auch die Qualität der Informationen schwankt in der Praxis. Genauigkeit und Korrektheit der Informationen sind teils unterschiedlich. Auch die Vertrauenswürdigkeit und Aktualität der Akquirierungspunkte ist verschieden.

		<b>Konzeptionell</b>	
		Primär	Sekundär
<b>Betrieblich</b>	Zeit	NTP-Zeitstempel	Auslastungsvorhersage basierend auf aktuellem Verkehrsaufkommen
	Ort	IP-Adresse	Signalstärken von Hotspots oder Metadaten versendeter Dateien
	Identität	MAC-Adresse einer Entität	Beziehungen zwischen Entitäten
	Aktivität	Port eines Prozesses	Identifikation einer Tätigkeit anhand laufender Prozesse
	Grund		Ableiten weshalb eine Aktion ausgeführt wurde

Abbildung 6.3: Placeholder

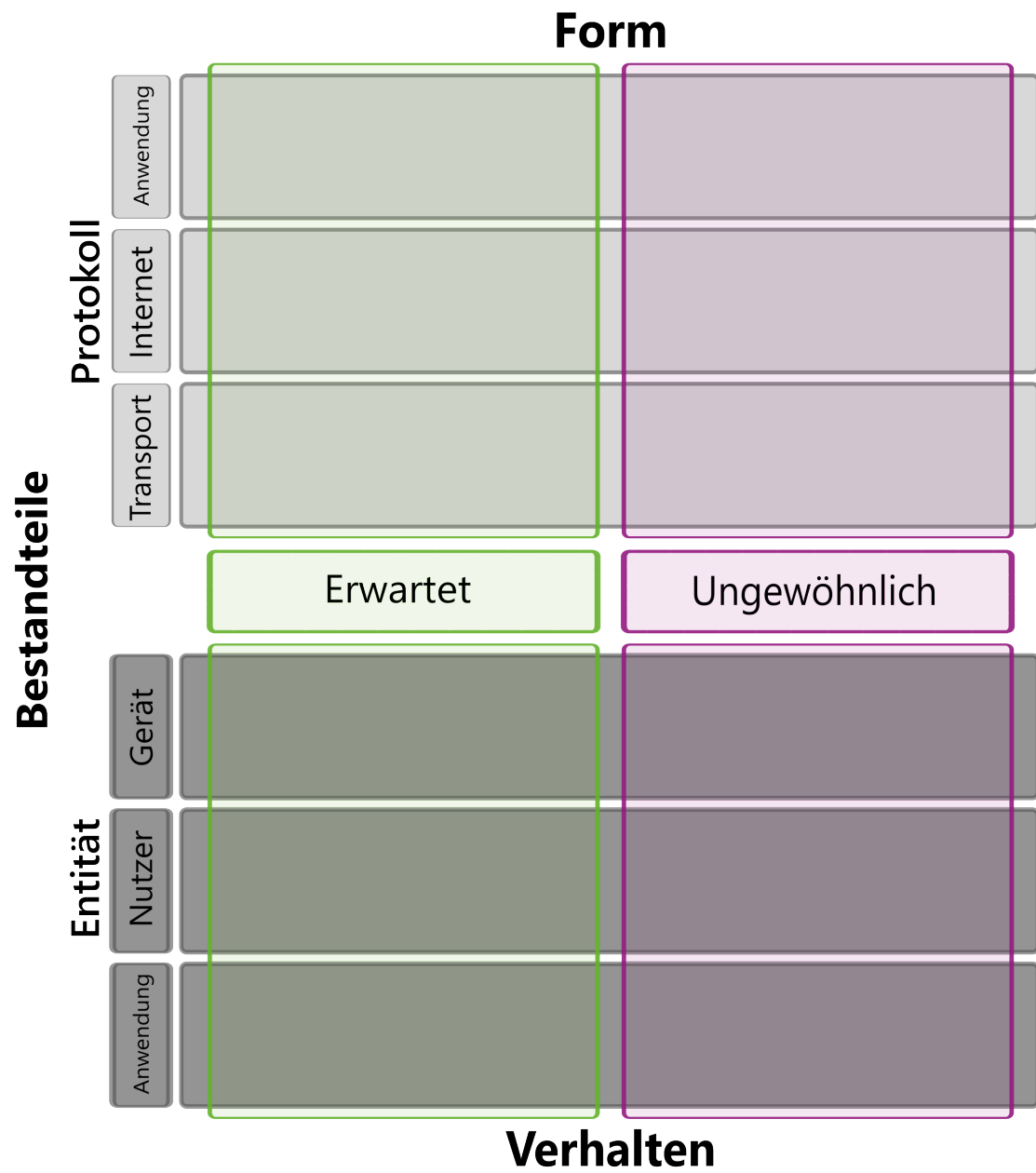


Abbildung 6.4: Placeholder

### 6.3 Leistungsverbesserung

# 7 Schlussfolgerung und Ausblick

## 7.1 Dateilose Prozesse

Die vorgestellte Implementation bezieht im besten Fall lediglich Prozesse die in ihrem Lebenszyklus auf die Festplatte schreiben mit ein und versucht dies zu Netzwerkverkehr zuzuordnen. Prozesse die nur im Arbeitsspeicher existieren werden nicht betrachtet sind aber ein zunehmendes Problem.

# Literatur

- [1] *About Zeek — Book of Zeek (git/master)*. URL: <https://docs.zeek.org/en/master/about.html> (besucht am 21.09.2022).
- [2] Gregory D. Abowd u. a. "Towards a Better Understanding of Context and Context-Awareness". In: *Handheld and Ubiquitous Computing*. Hrsg. von Hans-W. Gellersen. Bearb. von Gerhard Goos, Juris Hartmanis und Jan van Leeuwen. Bd. 1707. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, S. 304–307. ISBN: 978-3-540-66550-2 978-3-540-48157-7. DOI: 10.1007/3-540-48157-5\_29.
- [3] Jose Aguilar, Marxjhony Jerez und Taniana Rodríguez. "CAMEnto: Context awareness meta ontology modeling". en. In: *Applied Computing and Informatics* 14.2 (Juli 2018), S. 202–213. ISSN: 22108327. DOI: 10.1016/j.aci.2017.08.001.
- [4] Unai Alegre, Juan Carlos Augusto und Tony Clark. "Engineering context-aware systems and applications: A survey". en. In: *Journal of Systems and Software* 117 (Juli 2016), S. 55–83. ISSN: 01641212. DOI: 10.1016/j.jss.2016.02.010.
- [5] Mary Bazire und Patrick Brézillon. *Understanding Context Before Using It*. en. Hrsg. von David Hutchison u. a. Bd. 3554. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, S. 29–40. ISBN: 978-3-540-26924-3 978-3-540-31890-3. DOI: 10.1007/11508373\_3.
- [6] Suan Khai Chong u. a. "Context-Aware Sensors and Data Muling". In: (2007), S. 15.
- [7] John D Day und Hubert Zimmermann. "The OSI reference model". In: *Proceedings of the IEEE* 71.12 (1983), S. 1334–1340.
- [8] Anind K Dey. "Understanding and Using Context". In: (2001), S. 4.
- [9] Saul Greenberg. "Context as a dynamic construct". In: *Human-Computer Interaction* 16.2-4 (2001), S. 257–268.
- [10] Karen Henriksen. "A framework for context-aware pervasive computing applications". In: (2003).
- [11] A S M Kayes, Jun Han und Alan Colman. "ICAF: A Context-Aware Framework for Access Control". en. In: (2012), S. 8.
- [12] Peter Mell u. a. "An overview of issues in testing intrusion detection systems". In: (2003).
- [13] Aleksandar Milenkoski u. a. "Evaluating Computer Intrusion Detection Systems: A Survey of Common Practices". en. In: *ACM Computing Surveys* 48.1 (Sep. 2015), S. 1–41. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/2808691. (Besucht am 27.05.2022).

- [14] Charith Perera u. a. "Context Aware Computing for The Internet of Things: A Survey". In: *IEEE Communications Surveys & Tutorials* 16.1 (2014), S. 414–454. ISSN: 1553-877X. DOI: 10.1109/SURV.2013.042313.00197.
- [15] Karen Scarfone, Peter Mell u. a. "Guide to intrusion detection and prevention systems (idps)". In: *NIST special publication* 800.2007 (2007), S. 94.
- [16] Bill N Schilit, Norman Adams und Roy Want. "Context-Aware Computing Applications". In: (1994), S. 7.
- [17] Arthur H Van Bunningen, Ling Feng und Peter MG Apers. "Context for ubiquitous data management". In: *International Workshop on Ubiquitous Data Management*. IEEE. 2005, S. 17–24.
- [18] Wei Liu, Xue Li und Daoli Huang. "A survey on context awareness". In: *2011 International Conference on Computer Science and Service System (CSSS)*. 2011 International Conference on Computer Science and Service System (CSSS). Nanjing, China: IEEE, Juni 2011, S. 144–147. ISBN: 978-1-4244-9762-1. DOI: 10.1109/CSSS.2011.5972040.
- [19] J. Wolfgang Kaltz, Jürgen Ziegler und Steffen Lohmann. "Context-aware Web Engineering: Modeling and Applications". In: *Revue d'intelligence artificielle* 19.3 (1. Juni 2005), S. 439–458. ISSN: 0992499X. DOI: 10.3166/ria.19.439–458.
- [20] *zeek-agent-v2: Open source endpoint agent providing host information to Zeek. [v2]*. URL: <https://github.com/zeek/zeek-agent-v2> (besucht am 21.09.2022).
- [21] Andreas Zimmermann, Andreas Lorenz und Reinhard Oppermann. "An Operational Definition of Context". In: *Modeling and Using Context*. Hrsg. von Boicho Kokinov u. a. Bd. 4635. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, S. 558–571. ISBN: 978-3-540-74254-8. DOI: 10.1007/978-3-540-74255-5\_42.