IoT-Architektur zum Schutz von Privatsphäre Ende-zu-Ende

Sebastian Funke

TU Darmstadt · AGT International
sebastian.funke.mail@gmail.com

Zusammenfassung

Diese Arbeit stellt eine ganzheitliche Lösung zur Umsetzung von Privatheit im Internet der Dinge (IoT) vor. Wir erläutern die fünf Hauptbestandteile: ein IoT Referenz- und -Bedrohungsmodell, ein kompositorisches Privacy-Enhancing-Technology-Taxonomiemodell (PET-Taxonomiemodell) mit Privatsphäre-Metrik, eine Kontext- und Policy-sensitive Architektur zum Schutz von Privatsphäre in IoT-Service-Infrastrukturen, eine Prototyp-Implementierung und eine empirische Evaluation.

1 Einführung, Bestandteile und Verwandte Arbeiten

Die mit dem IoT einhergehende ständige Übertragung, Speicherung und Verarbeitung identifizierender und verhaltensbezogener Informationen führt zu Datenkontrollverlust und erhöhter Bedrohung der Privatsphäre. Zum Beispiel Profiling und Tracking von Personen. Verunsicherung durch potentielle Überwachung und Diskriminierung steigt und somit sinkt das Vertrauen in und die Verbreitung von IoT-Technologien. Daher erfordert Interaktion zwischen IoT und unserem persönlichen Leben Schutz der Privatsphäre [DaWK15, ZiMW14].

Privacy Enhancing Technologies (PETs), d.h. Technologiebausteine, die gezielt zur Realisierung einzelner spezifischer Privatheitsziele eingesetzt werden können, sind bekannt. Beispiele für solche PETs sind Tor [DiMS04] oder Differential Privacy [Dwor06], zur Anonymisierung der Netzwerkkommunikation und Daten. Üblicherweise sind PETs komplex und auf konkrete Anwendungsfälle spezialisiert. Um ein ganzheitliches Privatheitskonzept umzusetzen, ist es oft nötig mehrere PETs einzusetzen. Allerdings ist bislang unklar wie PETs in heterogenen und Kontext-sensitiven IoT-Infrastrukturen genutzt, komponiert, und miteinander verglichen werden können. Diese Problematik wird in der vorliegenden Arbeit aufgegriffen und gelöst.

1.1 Bestandteile

Die Arbeit gliedert sich in folgende Bestandteile: Um Szenarien, Akteure und Bedrohungen konkret und effizient beschreiben zu können wird ein IoT-Referenz- und -Bedrohungsmodell eingeführt. Um verschiedene PETs zu beschreiben und anhand ihrer Charakteristiken zu vergleichen, z.B. Schutzziele und Bedrohungsmodelle, wird in dieser Arbeit eine bestehende PET-Taxonomie [HZNF15] erweitert und formalisiert. Anschließend wird eine PET-Algebra definiert, um Privatsphären-Eigenschaften von IT Systemen mit mehreren PETs zu beschreiben und abgedeckte Schutzziele mit einer Metrik zu messen. Mit der PET-Algebra als Basis wird eine modulare und verteilte Softwarearchitektur entworfen, welche auf komponierten PETs basiert, die mit Kontext-sensitiven Policies gesteuert werden können. Darauf aufbauend wird ein

P. Schartner · A. Baumann (Hrsg.) · D•A•CH Security 2017 · syssec (2017) 159-169.

Prototyp der Architektur (PAPI) für mobile IoT-Anwendungen implementiert. Abschließend wird der Prototyp und seine Privatsphäre-schützenden Eigenschaften in einem Fitness Tracking Anwendungsfall empirisch evaluiert.

1.2 Verwandte Arbeiten

Es existieren mehrere PET/Privatsphäre-Modelle und -Metriken [Swee02, PfHa08, BoPa11, BKMM+13]. Diese beschäftigen sich aber meist nur isoliert mit Anonymität im Sinne des Schutzzieles Ununterscheidbarkeit und sind daher nur in spezifischen Fällen anwendbar. In dieser Arbeit werden zusätzliche Schutzziele, z.B. Vertraulichkeit und Vertrauen, berücksichtigt und von PETs abgeleitet formalisiert. Heurix et al. präsentieren eine generische PET-Taxonomie [HZNF15] ohne sie zu formalisieren oder Komposition zu untersuchen. Sie dient als Basis für diese Arbeit und wird entsprechend erweitert und formalisiert.

Populäre IoT-Referenz-Architekturen (IoT-A, OpenIoT, etc.) schützen Privatsphäre meist nur mit kryptographischer Zugriffskontrolle und vernachlässigen Schutzziele wie z.B. Pseudonymität [VDLG+15]. Es wurden mehrere Privatsphäre Architekturen für unterschiedliche Anwendungsgebiete (z.B. Web und Ubiquitous/Pervasive Computing) vorgeschlagen [HoLa04, Spei09, Kolt10]. Entweder sie sind nicht anwendbar in IoT oder decken nicht alle Schutz-Domänen und -Ziele ab.

Lioudakis et al. stellen die Discreet Box vor: ein Nutzer-zentrischer, Behörden-kontrollierter, Kontext- und Policy-sensitiver „Privacy Broker“ mit Personal Data Store (PDS) [LKDK+07]. Im Gegensatz zur Discreet Box, hat die Architektur in dieser Arbeit keine Behördenabhängigkeit und zielt eher auf den Einsatz in Firmen-Infrastrukturen ab.

Henze et al. präsentieren UPECSI: eine Datenmodell-getriebene Privatsphäre Architektur für Cloud-basierende IoT-Anwendungen mit kryptographischer Zugriffskontrolle und „Transparenz by Design“ [HHKH+16]. Sie ist ähnlich der hier vorgestellten Architektur, erfordert aber tiefgreifende Code-Änderungen (Model-Annotationen) vor Integration in bestehende Services. Weiterhin ermöglicht der PET-Modul Ansatz, der hier vorgestellten Arbeit, flexiblere Erweiterung für zukünftige Schutzbedürfnisse.

Eine modulare PET-Architektur für IoT, wie in dieser Arbeit vorgeschlagen, welche die Privatsphäre je nach Benutzerkontext über alle Schutz-Domänen hinweg schützt, ist nicht bekannt.

2 IoT Referenz- und Bedrohungsmodell

Abbildung 1 zeigt das angenommene IoT-Referenz- und -Bedrohungsmodell mit Privatsphäre-kritischem Datenaustausch zwischen drei Akteur-Gruppen: Kunden, Service Dienstleistern und nicht vertrauenswürdige dritte Parteien. IoT-Servicebereitstellung erfolgt durch Interaktion der Akteure, mit heterogenen Technologien über drei (Schutz-) Domänen: Smart Things z.B.: Sensoren, rechnerisch schwache Geräte (CoD), Aktoren und Smartphones (D); Infrastrukturen z.B.: Geräte-Controller und Gateways (G), und Services z.B.: Analytik in rechnerisch starken Clouds. Jede Domäne birgt eigene Bedrohungen für Daten, insbesondere da rechtliche Schutzbedingungen an Dienstleister teilweise als ungenügend angesehen werden [FDWK+15]. Deshalb ist eine technische End-zu-End (E2E) Lösung von Datenquelle (Nutzergerät) über die Infrastruktur zum Ziel (Cloud-Service) nötig.

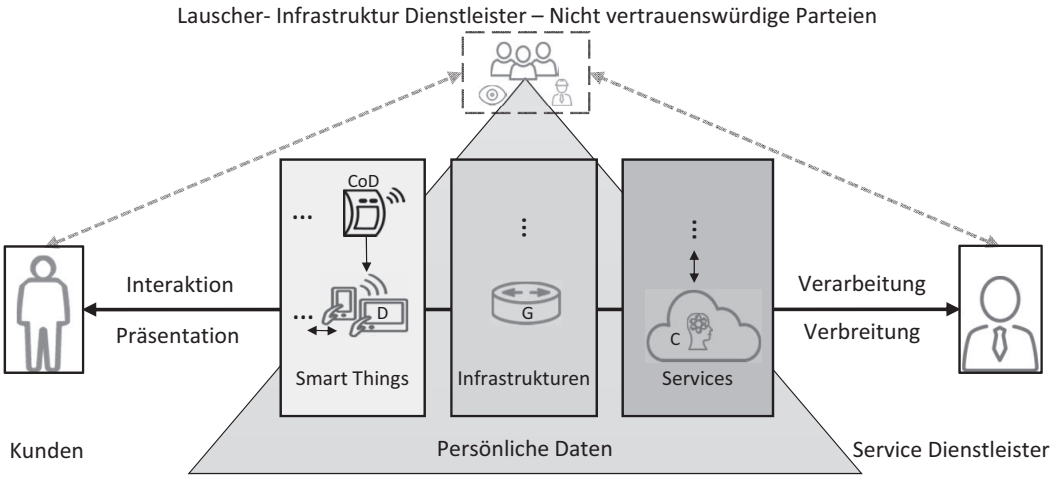

Abb. 1: Vereinfachtes IoT-Referenz- und -Bedrohungsmodell für die drei Akteure: Kunden, Service Dienstleister und nicht vertrauenswürdige dritte Parteien, mit Datenaustausch zwischen Entitäten in den Schutzdomänen: Smart Things, Infrastrukturen und Services

3 PET-Taxonomie-Modell, Komposition und Metrik

Um PETs formal zu beschreiben, zu vergleichen und zu komponieren ist ein formales Modell nötig. Als Basis für das PET-Modell dient die Taxonomie von Heurix et al. [HZNF15]. Der Taxonomiebaum enthält PET-Charakteristiken (Blätter) in sieben Dimensionen (Zweige): *Aim/Goal*, *Aspect*, *Scenario*, *Data*, *Foundation*, *TTP* und *Reversibility*.

3.1 PET-Taxonomie-Erweiterung und -Modellierung

Um ein mathematisch und semantisch korrektes Modell abzuleiten, auf dem numerische Metriken definiert werden können, werden u.a. Schutzziele erweitert (*Goal*-Charakteristiken), mit Stärken versehen ($\mathbb{Q}$) und semantisch doppelte Charakteristiken entfernt. Abbildung 2 zeigt die Erweiterungen (graue Boxen) und Modifizierungen (durchgestrichene Boxen) der Taxonomie. Nun werden die Dimensionen mathematisch als Mengen dargestellt und die Taxonomie als 8-Tupel mit diesen Mengen als Komponenten definiert:

$$\text{PET-Taxonomie-Model } PTM = (P, G, D, R, A, F, T, S)$$

Die *Goal*-Dimension G wird z.B. als Menge der möglichen PET-Schutzziele (*Unlinkability*, *Confidentiality*, etc.) modelliert, jeweils mit einer numerischen Stärke. Die Schutzstärken werden mit einem offenen Intervall $(0, 1)$[1] modelliert. Einzelne Elemente werden durch abgekürzte Subskript-Label identifiziert und einer Schutzziel-Charakteristik zugeordnet.

$$\begin{aligned} G = \{&STR_{\text{Aw}}, STR_{\text{Co}}, STR_{\text{De}}, STR_{\text{In}}, \\ &STR_{\text{Tr}}, STR_{\text{Ul.Pa}}, STR_{\text{Ul.Us}}\} \\ STR = \{&s \in \mathbb{Q} : s \in (0, 1)\} \end{aligned}$$

[1] Stärke 0 wird mit der leeren Menge modelliert, 1 d.h. z.B. perfekte Ununterscheidbarkeit, kann nur approximiert werden und beliebig viele Stärkegrade dazwischen erlauben flexible Kategorisierung von PET-Stärken

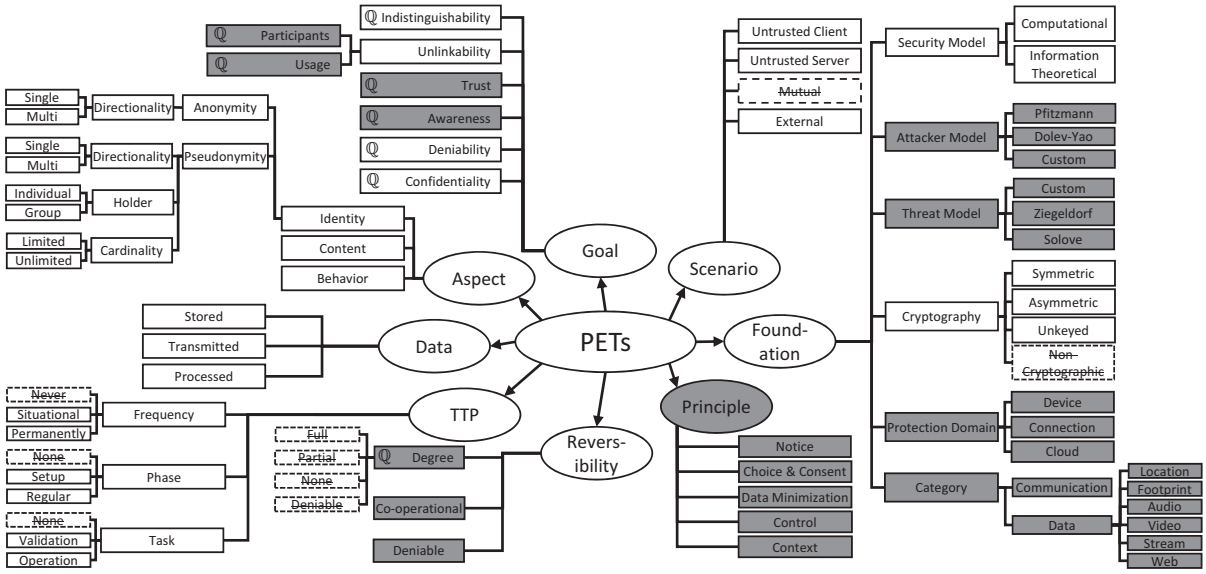

Abb. 2: Erweiterte PET-Taxonomie mit 8 Dimensionen, abgeleitet von [HZNF15]

Zum Beispiel, starke Netzwerkteilnehmer-Anonymität (*Participant Unlinkability – Ul.Pa*) eines MixNet PET's [DiMS04] kann subjektiv modelliert werden mit $G = \{0.99_{\text{Ul.Pa}}\}$. Mit welcher Stärke bestimmte PETs ein Schutzziel erfüllen oder welche PETs stärker sind als andere, hängt u.a. von ihrer Konfiguration und weiteren Charakteristiken ab. Darüber hinaus existieren formale Analyse-Frameworks, z.B. für *Unlinkability* [BKMM+13], mit denen, gegeben eines Angreifermodells und gewählten Parametern, die Schutz-Stärke 1 annehmen kann. Stärkenunterschiede von PETs könnten z.B. auch im Rahmen einer gemeinschaftlichen Maßnahme, mit Konsens zwischen politischen, wissenschaftlichen und industriellen Expertengruppen, ermittelt werden. Die möglichst genaue Bestimmung der Stärke einzelner PETs sprengt allerdings den Rahmen dieser Arbeit und ist Gegenstand von zukünftiger Forschung.

Der PET-Charakteristikraum (PTM') ergibt sich durch bilden der Potenzmenge der einzelnen Komponenten.

3.2 Kompositorische PET-Algebra

Für den späteren Anwendungsfall als Policy-Engine in der Architektur ist es nötig PETs mit einer abgeschlossenen, assoziativen und kommutativen Relation zu komponieren. Daher wird zunächst eine Algebra $\langle PTM', \oplus_c \rangle$ mit einem binären Operator $\oplus_c$ auf dem PET-Charakteristikraum PTM' definiert. Anwendungsfall-spezifische Einschränkungen werden als logische Prädikate (c) auf der Vereinigung ($\cup_c$) formuliert. So können zwei PETs $X, Y \in PTM'$ mit einer „Minimum-Schutz"-Einschränkung (w – Weakest Link Constraint) komponiert werden, bei der das resultierende PET-System die schwächeren Ziele gleicher Ziel-Kategorie annimmt. Zum Beispiel für zwei unterschiedliche Verschlüsselungssysteme, mit

verschieden starken Sicherheitsparametern (*Confidentiality* – Co), z.B. AES-128 und DES-56:

$$X_{\text{AES}} = \big(\ldots, \{0.8_{\text{Co}}, 0.2_{\text{Tr}}\}, \ldots\big)$$
$$Y_{\text{DES}} = \big(\ldots, \{0.5_{\text{Co}}\}, \ldots\big)$$
$$X \oplus_w Y = \big(\ldots, G_X \oplus_w G_Y, \ldots\big) = \big(\ldots, \{0.8_{\text{Co}}, 0.2_{\text{Tr}}\} \cup_w \{0.5_{\text{Co}}\}, \ldots\big) =$$
$$\big(\ldots, \{0.5_{\text{Co}}, 0.2_{\text{Tr}}\}, \ldots\big)$$

Assoziativität, Kommutativität und Abgeschlossenheit der Algebra wurden formal bewiesen.

3.3 Privatsphären-Schutz-Metrik für PET-Systeme

Basierend auf den Schutzzielen eines komponierten PET-Systems kann eine Privatsphären-Schutz Metrik *Privacy* : $PTM' \rightarrow [0, 1)$ berechnet werden:

$$\text{Privacy}(P) = \frac{\sum_{i=0}^{|g(P)|} s(x_i)}{|G|} \text{ mit Hilfsfunktionen } g : PTM' \rightarrow G' \text{ und } s : G \rightarrow STR$$

Die Metrik misst den Privatsphären-Schutz in einer Service Infrastruktur, relativ zu den darin angewandten PETs. Privacy$(P) = 0$ bedeutet dementsprechend, dass Privatsphäre in der IT Infrastruktur nicht durch PETs geschützt wird, bzw. PET-System P keine PETs mit Schutzzielen enthält. Privacy$(P) \approx 1$ steht für approximiert bestmöglich geschützte Privatsphäre relativ zu den eingesetzten PETs.

4 Architekturdesign und -beschreibung

Privatsphäre in IoT ist ein komplexes Problem mit kollidierenden Interessen verschiedener Akteure, welches eine Datenkontext-dynamische, Datenformat/Protokoll-unabhängige, verteilt-modulare, transparente, Kunden-unauffällige, skalierbare, zentral-steuerbare, erweiterbare und Policy-geschützte End-zu-End Lösung von Nutzergerät/Sensor bis Cloud-Service erfordert.

Die Architektur besteht aus zwei verteilten und entkoppelbaren PET-Framework-Middlewares, die in Kombination alle IoT-Schutzdomänen End-zu-End von Datenquelle bis Datenziel mit Privatsphäre schützenden PETs abdecken. Abbildung 3 zeigt das Architekturdesign in klassischer Client-Server-Perspektive mit Datenfluss (durchgezogene Pfeile) und Kontrollfluss (gestrichelte Pfeile) von Nutzergerät (Datenquelle links, z.B. Smartphone) zum End-Service (Datenziel rechts, z.B. Cloud-Service) durch die, mit PETs (gerahmte Boxen) ausgerüsteten, Middleware-Komponenten (gerahmte Boxen).

Privacy API (PAPI)

Framework/API auf ressourcenschwachen Smart Thing oder Infrastrukturgeräten (Smartphone, Geräte-Controller, Gateway, etc.) für „leichtgewichtige“ Daten- und Netzwerk-PETs, z.B. Daten-Whitelisting, Identität-Pseudonymisierung, Konsens-Dialoge/Benachrichtigungen, Mix/DC-Net/Crowd-Anonymisierung, Geo-Position-Verschleierung, etc. Darüber hinaus sind u.a. folgende funktionale Komponenten enthalten: Metadaten-Adapter, Gerät-Request-Interceptor, Kontext- und Policy-Resolver, sowie optionale Hilfskomponenten für bestimmte PETs (z.B. Trusted Third Party (TTP) für Pseudonymisierung).

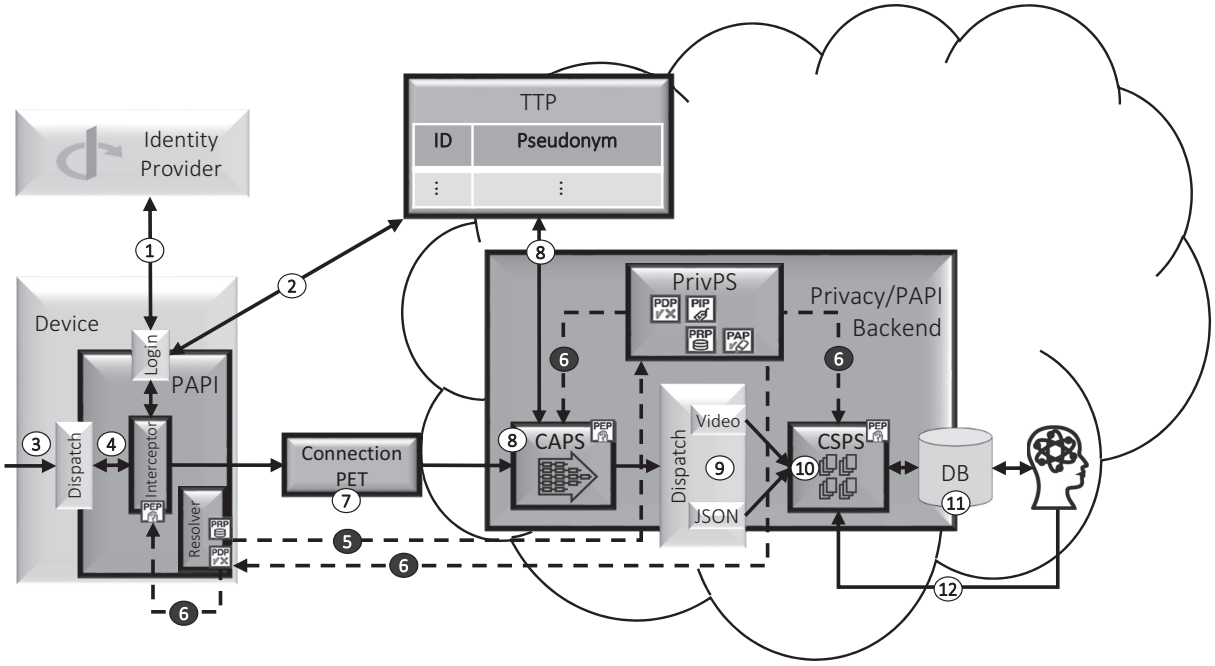

Abb. 3: Design der Architektur zum Schutz der Privatsphäre in üblichen IoT-Infrastrukturen

PAPI/Privacy Backend

Zentrale Verwaltungseinheit in Service/Cloud-Domäne mit Framework/API für „schwergewichtige“ und Daten/Format-spezifische PETs (Content Agnostic Privacy Service (CAPS) und Content Specific Privacy Service (CSPS)), z.B. Gesichts- und Stimmverschleierung in Video/Audio-Daten, Differential-anonymisierte Datenspeicherung [Dwor06], Privatsphäre-erhaltende Analytik wie Daten-Fusion/Aggregation etc., Datenvertrauens-Metriken, sowie Soft-/Hardware-Zustandsattestierung von Datenquellen [ISTZ16]. Weiterhin enthalten, ist die zentrale Policy-, Kontext-, Konfigurations- und PET-Verwaltung – Privacy Policy/Profile Service (PrivPS) – mit Kunden/Dienstleister/Entwickler/Behörden-Schnittstelle für die Verhandlung, Überprüfung und Änderung von Kontext-gebundenen PET-Konfigurations-Policies bzw. abgeleiteten kontextuellen Datenschutzrichtlinien – sowie der Backend-Request-Interceptor und optional eine verschlüsselte PDS-Datenbank [LKDK+07].

Daten und Kontrollfluss

Es folgt eine Schritt für Schritt Erläuterung des Daten- und Kontrollflusses (Abbildung 3, Schritte 1-12) am Beispiel des Fitness-Tracking-Anwendungsfalls. PAPI ist in diesem Beispiel als Android-Anwendungsbibliothek (HTTP Client Service) implementiert und das Backend als Web-Service mit transparenter Proxyfunktion.

Die Erstinstallation des Geräts/der Anwendung beginnt mit der Authentifikation des Benutzers (1), wobei optional die Benutzer-ID mit TTP pseudonymisiert (2) wird. Das Smartphone zeichnet auf und sendet Daten (3). Dabei werden sie von PAPI abgefangen und im Metadaten-Adapter (Dispatch) mit Kontrollattributen, für die PETs, annotiert (4). PAPI's Kontext-Resolver (5) ermittelt zu der Anfrage eine Benutzer/Geräte-Kontext-Instanz (Privacy Context Instance (PCI)), z.B. Zeitpunkt, Geo-Position und Benutzer-ID. Diese wird im Policy-Resolver auf eine PET-Konfigurationspolicy abgebildet. Eine Policy besteht aus einer Menge von PETs-

Konfigurationsparametern und Privacy Context Descriptions (PCDs), z.B. Zeitraum, Geo-Positionsraum und Benutzergruppe, auf welche die PCIs abgebildet werden können. Falls keine passende Policy lokal gefunden werden kann, wird die PCI passende Policy im Backend (PrivPS) ermittelt. Nun werden die PETs in PAPI, CAPS und CSPS, entsprechend der Policy konfiguriert (6), auf die annotierten Daten angewendet und weitergeleitet (7) zum PAPI Backend. Im Backend werden Daten- und Format-agnostische PETs angewandt, z.B. die Validität der Pseudonym-Token (8) bei der TTP- und Daten-Aggregation. Darauffolgend separiert der Format-Dispatcher (9) die Daten (Video, Audio, JSON, etc.) für Daten/Format-spezifische PETs (10), z.B. Video/Audio-Gesicht/Stimmverschleierung. Abschließend werden die, von PAPI angefügten, PET-Kontrollattribute entfernt, die verarbeiteten Daten werden zum Ziel-Service/Datenbank (11) weitergeleitet und Service-Antworten werden direkt über das Backend und PAPI zurückgegeben. Optional können die Backend PETs auch unabhängig vom Request-Interceptor als Privacy-Services (12) verwendet werden. Das Backend erlaubt die Kontext-spezifische Verhandlung von Policies zwischen autorisierten Kunden und Service-Mitarbeitern mit flexibler Zuordnung von PCDs und PET-Konfigurationsprofilen zu den Policies.

5 Prototyp Implementierung für Mobile IoT

Eine Instanz der Architektur wurde als Prototyp für Smartphone basierende IoT-Anwendungen implementiert – konkretisiert dargestellt in Abbildung 4. Der Prototyp enthält die zwei Haupt-

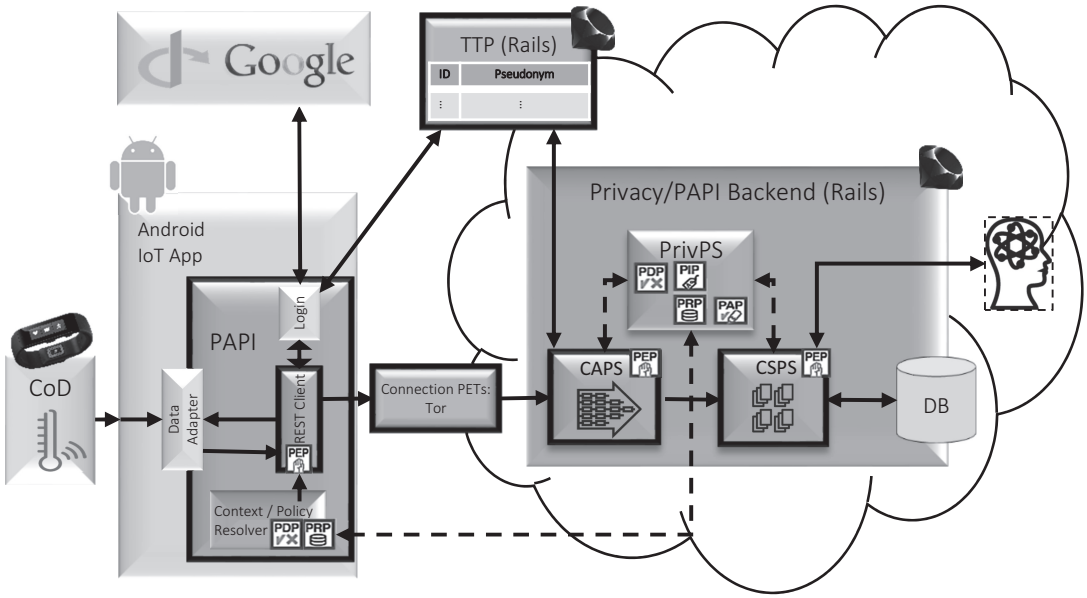

Abb. 4: Prototyp der Architektur, implementiert als Android-REST-Bibliothek (PAPI) und Ruby on Rails Web Service (Backend) für den Anwendungsfall Fitness-Tracking (Microsoft Band, links)

komponenten: PAPI (Bibliothek für Android-Anwendungen, links) und PAPI Backend (Web Service, rechts) – jeweils mit mehreren PETs und weiteren Sub-Komponenten. Die Anforderungen sind eine Untermenge der Architekturanforderungen und können in der Masterthesis nachgeschlagen werden.

Privacy API (PAPI)

PAPI (auf dem Mobiltelefon) wurde als HTTP-REST-Client-Bibliothek implementiert mit erweiterbarer API, Google als Identitätsprovider, einem JSON-De/Serialisierer als Metadaten-Adapter, ein Kontext-Generator/Resolver der Standard Android APIs nutzt und drei PETs: TTP-basierte Token-Pseudonymisierung, Tor Onion Routing (Orbot) und ein Daten-Whitelister. Dieser Bibliothek-Ansatz ist zwar schwieriger in bestehende Android-Anwendungen zu integrieren, erlaubt aber mehr Flexibilität für den Prototypen.

PAPI/Privacy Backend

Das PAPI Backend (sowie die TTP) wurde als Ruby on Rails (RoR) Web-Service implementiert mit zwei PETs: ein TTP-Token-Verifizierer und ein flexibler Daten-Perturbator zur feingranular steuerbaren Randomisierung verschiedener Datentypen, z.B. GPS-Koordinaten, Kreditkartennr., E-Mails und numerische Daten. Die Backend PETs können flexibel ausgetauscht, verwaltet und versioniert werden mit dem GEM-Paketmanager in RoR. PAPI PETs können ebenfalls darin versioniert werden. Die automatisierte Auslieferung und Synchronisation von geändertem PET-Source-Code, zwischen Backend's PET-Register und den ausgerollten PAPI Bibliotheken, auf Kunden-Smartphones, wurde für zukünftige Verbesserung offen gelassen. Verarbeitete Daten werden zum Vergleich für die spätere Evaluation in einer lokalen Datenbank gespeichert. Die Benutzerschnittstelle, für autorisierte Service-Mitarbeiter, erlaubt die Verwaltung der Policies, PCDs, PET-Konfigurationen und Konfigurationsprofile. Weiterhin wird jedes PET mit einer Instanz des PET-Taxonomiebaums (PTM) ausgeliefert, welcher ebenfalls im Backend geändert werden kann und u.a. zur Berechnung der Privatsphären-Schutz-Metrik, für erstellte Policies, genutzt wird.

6 Quantitative und Qualitative Evaluation

Die Privatsphäre-kritische Quantified-Self-Bewegung (QS-Bewegung) wird immer populärer für Privatnutzer und Service-Dienstleister, z.B. Tracking von Fitness-Daten für Gesundheitsversicherungen und Sportvereine. Damit einhergehend, steigt das Tracking-Risiko von Verhaltens- und Gesundheits-bezogenen Daten. Somit ist Fitness-Tracking ein idealer Anwendungsfall zur Evaluation von PAPI.

Es wurden zwei Ansätze gewählt: ein Experteninterview zur qualitativen Bewertung des Privatsphären-Schutzes und eine quantitativ, experimentelle Machbarkeitsstudie der Architektur mithilfe des Prototyps. Im Fokus der Evaluation stand die Privatsphären-Sensitivität und Informationsmenge (Entropie) aufgezeichneter Fitnessdaten in verschiedenen Kontexten (während oder nach der Arbeitszeit, in bestimmten Umgebungen, etc.) und wie sich diese durch den Einsatz von PAPI ändert.

Quantitativ, Experimentelle Evaluation

Im Experiment trugen die Teilnehmer ein Microsoft Band und ein Smartphone mit Daten sammelnder Fitness Anwendung – während sie acht festgelegte, fünf-minütige und aufeinanderfolgende Alltagsaktivitäten (z.B. Meeting, Indoor/Outdoor-Sport und Mittagessen) verrichteten. Dabei fand jede Aktivität in einem festgelegten Kontext, mit unterschiedlichen Privatsphäre-Risiken, statt, z.B. essen zur Mittagszeit in der Firma, ist weniger kritisch, als Outdoor-Sport während der Arbeitszeit. PAPI wurde in die Fitness-Anwendung integriert, das Backend und die TTP wurden auf zwei Servern installiert. Für die Risiko-variierenden Kontexte wurden vorher

im Backend entsprechende PET-Konfigurationsprofile und Policies eingerichtet. Anschließend wurden die gesammelten Daten (JSON) jeweils einmal mit und ohne aktivierter PAPI Middleware zur Zieldatenbank gesendet.

Um eine grobe quantitative Aussage über Unterschiede der Informationsmenge und zum Teil auch der Datensensitivität zu machen – mit und ohne PAPI – wurde JSON Key/Value Wort-Entropie (J) als Metrik gewählt. Abbildung 5 fasst die Entropie-Unterschiede der einzelnen Aktivitätsdatenspuren mit (links, dunkel) und ohne PAPI (rechts, hell) zusammen. Erwartungs-

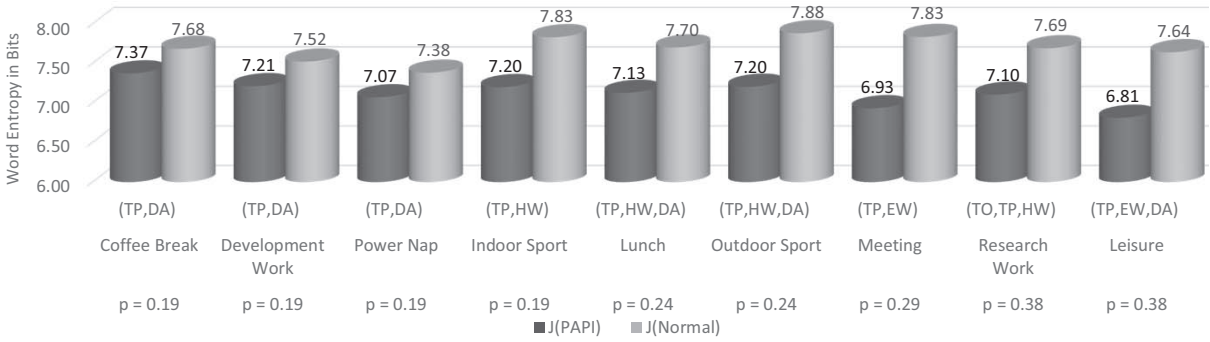

Abb. 5: JSON-Wort Entropie (J in Bits) der Aktivitätsdaten mit (links, dunkel) und ohne PAPI (rechts, hell), in 9 verschiedenen Kontexten, sortiert aufsteigend nach Schutzstärke (p) von links nach rechts der zum Kontext aufgelösten Konfigurations-Policy

gemäß enthalten die, durch PAPI getunnelten, Daten weniger Informationen, im Durchschnitt 7,4%. Aktivitäten mit Privatsphäre-kritischem Kontext, z.B. Freizeit ($p = 0.38$) und Meetings ($p = 0.29$) – dementsprechend Policies mit restriktiv konfigurierten PETs – erreichen eine maximale Entropie-Differenz von 11.5%. Weiterhin ergab die Entropie-Messung, dass die einzelnen, von PAPI verarbeiteten, Datenspuren im Durchschnitt 32% größer (in Bits) sind, aufgrund der zusätzlichen Kontrollattribute, eingefügt vom Metadaten-Adapter. Diese Hilfslabel werden allerdings in der Backend Middleware wieder entfernt, sodass PAPI nur ein geringen Overhead verursacht in der Netzwerkauslastung zwischen PAPI und Backend. Die PETs wurden aus Zeitbeschränkung der Thesis nicht im Einzelnen evaluiert und Entropie ist keine ausreichende Metrik zur Beurteilung von Privatsphäre, daher wurde zusätzlich das Experteninterview veranstaltet.

Qualitative Experteninterview Evaluation

Das Experteninterview hatte zum Ziel die, durch PAPI, reduzierte Fitnessdaten-Sensitivität, mithilfe interaktiver Fragen an Experten, zu ermitteln. Dabei wurde ein IoT-Service-Experte aus der Industrie und ein PET/Datenschutz-Experte aus dem akademischen Umfeld zu den visualisierten End-Daten des Fitness-Tracking-Experiments (mit und ohne PAPI) befragt. Um die Privatsphäre-schützenden Eigenschaften der eingesetzten PETs unter bestimmten Kontexten zu evaluieren, wurden den Experten u.a. folgende Fragen gestellt:

Welche der Kontexte sind für die Privatsphäre/Vertraulichkeit des Kunden/Arbeitgebers am kritischsten? Welche Datenkategorien sind in den Datenspuren enthalten und wie sensitiv sind diese? Welche Datenspur (mit oder ohne PAPI) enthält mehr Informationen? Wie sensitiv sind die gesammelten Daten (z.B. Herzschlagverlauf und Gyroskop-Diagramm) mit und ohne PAPI?

Die Experten identifizierten und beurteilten die, durch PAPI, reduzierte Datensensitivität in

kritischen Kontexten positiv und äußerten verbleibende Herausforderungen und Bedrohungen, u.a. in Bezug auf Kontext-Tracking und reduzierter Service-Leistung. Experten werten sensitive Kontexte als Privatsphäre-kritisch, wenn diese einem Nutzer zugeordnet werden können. Beispielsweise könnte ein Dienstleister über Korrelation von Datenänderungen (durch PAPI) und Kontextraum, den sensitiven Kunden-Kontext ermitteln und somit Kontext-Profiling und -Tracking durchführen. Inwiefern PAPI die Service-Leistung negativ beeinflusst war nicht Gegenstand der Evaluation und hängt stark von den eingesetzten PETs ab. Diese Risiken müssen in weiteren Fallstudien erforscht werden um geeignete Risikominimierungsmaßnahmen zu finden.

7 Zusammenfassung und Ausblick

PAPI ist eine erweiterbare, verteilte, kontextuelle Policy- und PET-Framework basierende, Privatsphäre schützende Middleware für IoT-Firmeninfrastrukturen. Das Taxonomie-Modell (PTM) ist das Herzstück der Architektur und ermöglicht komponierbare PET Policies zur flexiblen Konfiguration des Privatsphären-Schutzes. Die Machbarkeitsstudie bestätigt, dass Privatsphäre in der Kontext-sensitiven IoT-Welt technisch gemessen und geschützt werden kann.

Diese Arbeit ebnet den Weg für weiterführende Arbeiten. Zum Beispiel die Messung von verbleibenden und konkreten Privatsphäre-Risiken (Inverse Schutzziele), Simulation von technischer PET-Komponierbarkeit, die Erforschung von Privatsphäre-Kontexten und die automatisierte Übersetzung zwischen technischen und rechtlichen Datenschutzrichtlinien. Das evaluierende Experiment und Experteninterview hat die Machbarkeit und Privatsphäre-schützenden Eigenschaften der Architektur bestätigt, aber nichtsdestotrotz verbleiben einige Herausforderungen. Tracking und Profiling von Kunden-Kontexten muss vermieden werden. Außerdem sollte bei der Verhandlung von Policies, die optimale Balance zwischen Kunden-Privatsphäre, Service-Leistung und Service-Sicherheit gewährleistet werden.

Literatur

[BKMM+13] M. Backes, A. Kate, P. Manoharan, S. Meiser, E. Mohammadi: AnoA: A framework for analyzing anonymous communication protocols. *In: Computer Security Foundations Symposium (CSF), 2013 IEEE 26th*, IEEE (2013), 163–178.

[BoPa11] J.-M. Bohli, A. Pashalidis: Relations Among Privacy Notions. In: *ACM Trans. Inf. Syst. Secur.*, 14, 1 (2011), 4:1–4:24.

[DaWK15] J. Daubert, A. Wiesmaier, P. Kikiras: A view on privacy & trust in IoT. *In: Communication Workshop (ICCW), 2015 IEEE International Conference on Communications*, IEEE (2015), 2665–2670.

[DiMS04] R. Dingledine, N. Mathewson, P. Syverson: Tor: The Second-generation Onion Router. *In: Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, USENIX Association (2004), 21–21.

[Dwor06] C. Dwork: Differential privacy. *In: Automata, languages and programming*, Springer (2006), 1–12.

[FDWK+15] S. Funke, J. Daubert, A. Wiesmaier, P. Kikiras, M. Muehlhaeuser: End-2-End privacy architecture for IoT. *In: Communications and Network Security (CNS), 2015 IEEE Conference on*, IEEE (2015), 705–706.

[HHKH+16] M. Henze, L. Hermerschmidt, D. Kerpen, R. Häußling, B. Rumpe, K. Wehrle: A comprehensive approach to privacy in the cloud-based Internet of Things. In: *Future Generation Computer Systems*, 56 (2016), 701–718.

[HoLa04] J. I. Hong, J. A. Landay: An Architecture for Privacy-sensitive Ubiquitous Computing. *In: Proceedings of the 2Nd International Conference on Mobile Systems, Applications, and Services*, MobiSys '04 (2004), 177–189.

[HZNF15] J. Heurix, P. Zimmermann, T. Neubauer, S. Fenz: A taxonomy for privacy enhancing technologies. In: *Computers & Security*, 53 (2015), 1 – 17.

[ISTZ16] A. Ibrahim, A.-R. Sadeghi, G. Tsudik, S. Zeitouni: DARPA: Device attestation resilient to physical attacks. *In: Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, ACM (2016), 171–182.

[Kolt10] J. Kolter: User-Centric Privacy, *Electronic Commerce*, Bd. 41. EUL-Verlag, Lohmar (2010), , zugl.: Regensburg, Univ., Diss., 2009.

[LKDK+07] G. V. Lioudakis, E. A. Koutsoloukas, N. Dellas, S. Kapellaki, G. N. Prezerakos, D. I. Kaklamani, I. S. Venieris: A proxy for privacy: the discreet box. *In: EUROCON, 2007. The International Conference on "Computer as a Tool"*, IEEE (2007), 966–973.

[PfHa08] A. Pfitzmann, M. Hansen: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management – A consolidated proposal for terminology. In: *Version v0*, 31 (2008), 15.

[Spei09] R. Speicys Cardoso: A Service-oriented Middleware for Privacy Protection in Pervasive Computing. Thesis, Université Pierre et Marie Curie - ParisVI (2009).

[Swee02] L. Sweeney: k-anonymity: A model for protecting privacy. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10, 05 (2002), 557–570.

[VDLG+15] E. Vasilomanolakis, J. Daubert, M. Luthra, V. Gazis, A. Wiesmaier, P. Kikiras: On the Security and Privacy of Internet of Things Architectures and Systems. *In: Secure Internet of Things (SIoT), 2015 International Workshop on*, IEEE (2015), 49–57.

[ZiMW14] J. H. Ziegeldorf, O. G. Morchon, K. Wehrle: Privacy in the Internet of Things: threats and challenges. In: *Security and Communication Networks*, 7, 12 (2014), 2728–2742.