Autorisierungsmanagement für das Internet of Things

Stefanie Gerdes[1] · René Hummen[2] · Olaf Bergmann[1]

Universität Bremen[1]
{gerdes | bergmann}@tzi.org

RWTH Aachen[2]
hummen@comsys.rwth-aachen.de

Zusammenfassung

Fortschritte im Bereich der Informationstechnik führen zur Entwicklung immer kleinerer und kostengünstigerer Mikroprozessoren. Dies erlaubt es, Alltagsgegenstände, Gebäudeinstallationen oder auch Maschinenteile in Produktionsanlagen mit Rechensystemen auszustatten. Solche „Smart Objects“ werden typischerweise speziell für einen bestimmten Zweck entwickelt, haben oft neben der Netzanbindung keine weiteren Nutzungsschnittstellen und können in der Regel nur einzelne, einfache Aufgaben erfüllen. Die Vernetzung von Smart Objects ermöglicht die Kommunikation dieser Geräte untereinander sowie mit deren Besitzern und eröffnet eine Vielzahl neuer Anwendungsfelder, die heute unter Schlagwörtern wie *Internet of Things* oder *Industrie 4.0* allgegenwärtig sind. Dabei wird erwartet, dass Smart Objects in alle Bereiche des täglichen Lebens integriert werden und eine Vielzahl personenbezogener oder anderweitig sensibler Daten verarbeiten. Die Ressourcenbeschränkungen der eingesetzten eingebetteten Systeme erschweren allerdings die Nutzung etablierter Sicherheitsmechanismen. Diese Beobachtung trifft ebenfalls auf Autorisierungsmechanismen zu, welche Nutzern die Möglichkeit geben, festzulegen, wer welche Aktionen auf welchen Daten durchführen darf. Eine bisher ungeklärte Frage betrifft die sichere Konfigurierbarkeit von Autorisierungen im Lebenszyklus von Smart Objects, insbesondere unter Berücksichtigung sich ändernder Besitzverhältnisse. Der vorliegende Beitrag zeigt auf, welche Maßnahmen ergriffen werden müssen, um den Schutz der Autorisierungsinformationen im gesamten Lebenszyklus eines Smart Objects sicherzustellen und stellt eine mögliche Lösung am Beispiel des Protokolls DCAF vor.

1 Einleitung

Smart Objects sind vernetzte Geräte aus der physischen Welt, die dazu entwickelt werden, einzelne, einfache Aufgaben zu erfüllen. Eine wichtige Voraussetzung für den breiten Einsatz dieser Geräte ist, dass sie kostengünstig hergestellt werden können. Dies führt oft dazu, dass ihre System-Ressourcen sehr begrenzt sind und ihre Leistungsfähigkeit stark eingeschränkt ist. Insbesondere im Kontext eingebetteter Systeme verfügen Smart Objects daher häufig über nur geringe Prozessorleistung und Übertragungskapazitäten, sind oft batteriebetrieben und bieten oftmals keine dedizierten Nutzungsschnittstellen wie z.B. Tastaturen oder Displays.

Es wird erwartet, dass die Anzahl der Smart Objects in Zukunft rapide ansteigen wird. Bis 2020 wird laut Gartner ein Anstieg auf 26 Mrd. von an das Internet angeschlossenen Geräten erwar-

P. Schartner · K. Lemke-Rust · M. Ullmann (Hrsg.) · D•A•CH Security 2015 · syssec (2015) 397-408.

tet.[1] Smart Objects werden von Moore's Law [Moor65] anders beeinflusst als heute gängige Systeme. Während sich die bisherige Entwicklungsarbeit darauf konzentrierte, leistungsstärkere und schnellere Prozessoren herzustellen sowie die verfügbare Speicherausstattung stetig zu erhöhen, liegt der Fokus im Internet of Things (IoT) darauf, die dort eingesetzten Geräte immer kostengünstiger und kleiner zu machen. Im Rahmen vieler Anwendungsfelder ist daher nicht zu erwarten, dass sich die Leistungsfähigkeit von Smart Objects innerhalb der nächsten Jahre signifikant steigern wird. Vielmehr ist von einer fortwährenden Miniaturisierung und Vergünstigung der eingesetzten eingebetteten Systeme auszugehen. Die weitreichende technische Bedeutung dieser Entwicklung manifestiert sich bereits heute in zahlreichen Standardisierungsaktivitäten, von Industriekonsortien wie der Open Mobile Alliance (OMA)[2] bis hin zum World Wide Web Consortium (W3C)[3] und der Internet Engineering Task Force (IETF)[4].

Smart Objects werden in alle Bereiche des täglichen Lebens integriert und erhalten dadurch eine Vielzahl personenbezogener oder anderweitig sensibler Daten. Um ihre Aufgaben optimal erfüllen zu können, müssen diese Geräte darüber hinaus häufig unbemerkt und autonom kommunizieren. Der Schutz von Smart Objects vor Angriffen aus dem Netz ist daher von großer Bedeutung, wird jedoch durch deren Ressourcenbeschränkungen enorm erschwert.

Durch aktuelle Protokollanpassungen [HWZH+13, HSRV+14, TsFo15] können bereits heute erprobte Sicherheitsprotokolle wie Datagram Transport Layer Security (DTLS) im Internet of Things eingesetzt werden. Allerdings gewährleisteten diese Sicherheitsprotokolle ausschließlich die *Authentizität*, *Integrität* und *Vertraulichkeit* der zwischen den kommunizierenden Parteien ausgetauschten Nachrichten. Die Frage nach der *Autorisierung*, also wer mit welchen Berechtigungen auf welche potenziell sensiblen Systemressourcen von Smart Objects zugreifen darf, ist jedoch noch nicht abschließend geklärt.

Ein essentieller Bestandteil einer umfassenden Autorisierungslösung im Internet of Things ist die sichere Vergabe von *Autorisierungsberechtigungen* für ein Smart Object. Nur so kann gewährleistet werden, dass ausschließlich autorisierte Nutzer, d.h. beispielsweise der Besitzer eines Geräts, über die Autorisierung von Aktionen bestimmen darf. Der Schutz der Autorisierungsberechtigung muss im gesamten Lebenszyklus eines Geräts lückenlos sichergestellt werden, um die Sicherheit der Informationen auf den Geräten nicht zu gefährden. Insbesondere Übergangsphasen im Lebenszyklus des Smart Objects (z.B. dessen Verkauf) erfordern ein sorgfältiges Design der Autorisierungslösung.

Der vorliegende Beitrag zeigt auf, welche Anforderungen eine Autorisierunglösung für das Internet of Things erfüllen muss, um den Schutz der Autorisierungsberechtigung im gesamten Lebenszyklus eines Smart Objects zu gewährleisten. Weiterhin wird eine entsprechende Lösung am Beispiel des Delegated CoAP Authentication and Authorization Frameworks (DCAF, [GeBB15]) vorgestellt. Diese Lösung basiert auf dem Constrained Application Protocol (CoAP, [ShHB14]) als zugrundeliegendem Übertragungsprotokoll, das speziell für die besonderen Anforderungen von Geräten mit eingeschränkten Systemressourcen entwickelt wurde und somit als Alternative zu HTTP im Internet of Things dient.

Die folgenden Abschnitte sind wie folgt strukturiert. Abschnitt 2 motiviert die Notwendigkeit für Autorisierungslösungen im Internet of Things. In Abschnitt 3 wird der Lebenszyklus der

[1] http://www.gartner.com/newsroom/id/2636073
[2] http://openmobilealliance.org/
[3] http://www.w3.org/
[4] https://www.ietf.org/

Geräte beschrieben. Abschnitt 4 analysiert bestehende Vorschläge für die Autorisierung von Zugriffen auf Smart Objects. Die nötigen Maßnahmen für das Autorisierungsmanagement im Laufe des Lebenszyklus werden in Abschnitt 5 eingeführt. Der Beitrag schließt mit einem Fazit sowie einem Ausblick in Abschnitt 6.

2 Herausforderungen für Autorisierung im IoT

Smart Objects lassen sich häufig einem einzelnen Eigentümer zuordnen. Je nach Einsatzgebiet kann es aber sein, dass mehrere Interessengruppen über das Verhalten des Smart Objects bestimmen dürfen. Dies ist zum Beispiel der Fall, wenn sich der Eigentümer und die Nutzer bzw. Besitzer eines Geräts unterscheiden. Dieser Beitrag beleuchtet, *welche* Herausforderungen sich bei der Übergabe der Konfigurationsverantwortung zwischen verschiedenen Parteien ergeben und *wie* diesen Herausforderungen begegnet werden kann. Die Frage, in wessen Verantwortlichkeit die Konfiguration eines Geräts für ein Mehrmandantensystem liegt, ist hingegen nicht Gegenstand dieses Beitrags. Daher wird in weiten Teilen auf eine differenzierte Unterscheidung zwischen Eigentümer und Besitzer verzichtet und stattdessen der Begriff „Nutzer“ verwendet.

Die Einsatzszenarien für Smart Objects sind vielfältig: Sie reichen von Heim- und Gebäudeautomation über die Steuerung von Industrieanlagen und Anwendungen im Gesundheitsbereich bis zur Beobachtung von verderblichen Gütern in Containern (vgl. [SGSM+15]). Je nach Einsatzgebiet haben die für ein Smart Object verantwortlichen Nutzer unterschiedliche Anforderungen an eine Autorisierungslösung. So ist im Gesundheitsbereich die Vertraulichkeit von Daten von großer Bedeutung, während es bei Industrieanlagen vor allem auf die Integrität der Daten ankommt. Gemein ist den Anwendungsgebieten, dass Nutzer davon profitieren, Autorisierungen für ihre Geräte und die darauf befindlichen Daten festlegen zu können.

Eine differenzierte Rechtevergabe erlaubt eine feingranulare Autorisierung von Zugriffen, d.h. jedes Gerät erhält nur die Berechtigungen, die es zur Erfüllung seiner Aufgabe benötigt. So lassen sich beispielsweise die Autorisierung zum Auslesen von Sensordaten und die Erlaubnis zum Ändern der Gerätekonfiguration voneinander trennen. Dies erschwert nicht nur die Manipulation des Smart Objects durch Unbefugte, sondern auch den Missbrauch eines Smart Objects als Einfallstor, um andere Geräte im Netz anzugreifen.

Moderne Web-Anwendungen setzen auf Autorisierungsmechanismen wie OAuth [Hard12] or OpenID [SBJM+14], die sich zum Zeitpunkt des Zugriffs auf eine geschützte Ressource an den Nutzer wenden, der die Anfrage veranlasst hat, in der Annahme, dass es sich dabei um den Eigentümer der zu schützenden Daten handelt.

Dieses Interaktionsmodell lässt sich allerdings nicht ohne weiteres auf das Internet of Things übertragen. Zum Einen verfügen Smart Objects zumeist nicht über geeignete Nutzungsschnittstellen, über die der betreffende Nutzer kontaktiert werden kann, um über den Zugriffswunsch zu entscheiden. Zum Anderen sollen vernetzte Smart Objects autonom agieren, ohne dass immer ein menschlicher Nutzer für Rückfragen bereitsteht. Kommunikationsendpunkte im Internet of Things müssen also in der Lage sein, Autorisierungsvorgaben selbständig durchzusetzen.

Je nach Art des Smart Objects ist deren Leistungsfähigkeit mehr oder weniger stark eingeschränkt. RFC 7228 [BoEK14] unterscheidet drei Klassen von Geräten. Die kleinste Klasse, die gesichert IP sprechen kann, ist Klasse 1 mit etwa 10 KiB RAM und um die 100 KiB nichtflüchtigem Speicher. Damit sind sie sehr viel weniger leistungsfähig als handelsübliche Smartphones oder auch Raspberry Pis.

Die technische Umsetzung von Autorisierungslösungen für das Internet of Things muss speziell die begrenzten Systemressourcen von Smart Objects berücksichtigen. Ein schlecht gestalteter Autorisierungsmechanismus kann berechtigten Anwendern den Zugriff verwehren und dadurch die Verfügbarkeit des von dem Gerät zur Verfügung gestellten Dienstes beeinträchtigen.

Das Fehlen von Nutzungsschnittstellen wie Tastaturen oder Displays führt dazu, dass die Konfiguration von Autorisierungsberechtigungen nicht am Gerät selbst vorgenommen werden kann, sondern die Netzschnittstelle verwendet werden muss, die auch für die übrige Kommunikation vorgesehen ist. Dieser Datenaustausch ist speziell zu sichern, um eine Kompromittierung des Smart Objects zu verhindern. Der Schutz der Autorisierungsinformationen ist in den Übergangsphasen im Lebenszyklus eines Smart Objects besonders schwierig, wie im folgenden Abschnitt dargestellt wird.

3 Lebenszyklus

Der Lebenszyklus eines Smart Objects kann vereinfacht in sechs Phasen unterteilt werden (siehe Abbildung 1). Er beginnt mit der Herstellung des Gerätes in der Herstellungsphase. Nach dem Verkauf durch den Hersteller folgt die Phase der Inbetriebnahme, in der Endnutzer die Smart Objects in Besitz nehmen und in ihr Netz integrieren. In der Betriebsphase erfüllen die Geräte ihren eigentlichen Zweck, interagieren also mit anderen Geräten und ihren Nutzern gemäß ihrer zugedachten Aufgabe. Die Betriebsphase wird gelegentlich von Wartungsphasen unterbrochen, in denen das Gerät repariert und gegebenenfalls neu konfiguriert oder seine Software auf den neuesten Stand gebracht wird. Darüber hinaus können Smart Objects den Besitzer wechseln, zum Beispiel weil sie verkauft oder verschenkt werden, oder auch, weil sie Teil eines Vermögens sind, das gerichtlich verfügt übereignet wurde. Diese Phase wird Übergabephase genannt. Der Lebenszyklus schließt mit der Außerbetriebnahme, bei der das Gerät entsorgt wird.

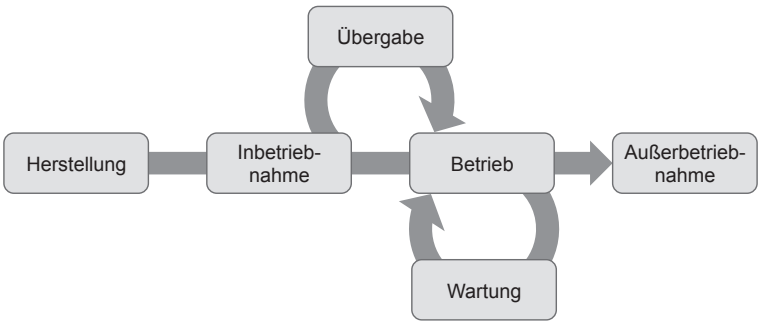


Abb. 1: Lebenszyklus eines Smart Objects

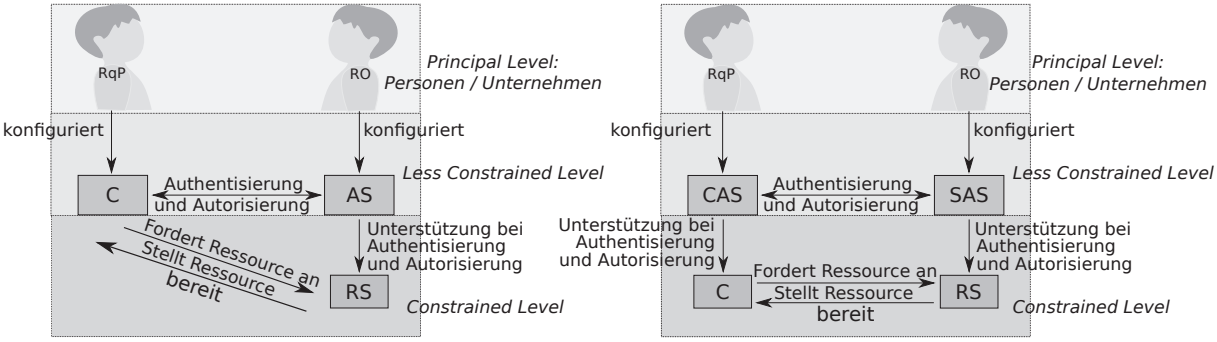
Eine besondere Herausforderung im Lebenszyklus stellen die Phasen dar, in der sich die Verantwortung für das Smart Object ändert. Vom Vorbesitzer konfigurierte Zugriffsberechtigungen müssen unwirksam werden. Darüber hinaus muss der neue Nutzer in die Lage versetzt werden, Zugriffsregeln zu definieren, während dem Vorbesitzer diese Berechtigung entzogen wird. Dieser Wechsel der *Autorisierungsberechtigung* ist besonders zu schützen, da derjenige, der diese Berechtigung besitzt, volle Kontrolle über das Gerät und die darauf befindlichen Daten hat.

Im Folgenden werden nun aktuelle Standardisierungsbemühungen im Kontext von Autorisierungslösungen für Smart Objects beschrieben. Es wird zunächst ein allgemeiner Überblick über den jeweiligen Lösungsvorschlag gegeben und anschließend auf die sichere Unterstützung der Übergangsphasen im Lebenszyklus eines Smart Objects fokussiert.

4 Aktuelle Standardisierungsbemühungen

Im Internet of Things spielen Autorisierungsserver (AS) eine wichtige Rolle, da sie die Verwaltung von Autorisierungen erleichtern können. Nutzer mit entsprechender Berechtigung (z.B. der Besitzer eines Geräts oder der Eigentümer sensibler Daten) konfigurieren vorab Zugriffsregeln für geschützte Ressourcen von Endpunkten, für die der betreffende AS zuständig ist. Auf der Basis dieser Konfigurationen kann ein AS automatisch entscheiden, wie und von wem auf welche Ressourcen zugegriffen werden darf. AS stellen somit das Bindeglied zwischen Nutzern und deren Geräten dar.

In der IETF wurde im Jahr 2014 die Arbeitsgruppe „Authentication and Authorization for Constrained Environments“ (ACE)[5] formiert, die einen Autorisierungsmechanismus standardisieren soll, der an die speziellen Anforderungen des Internet of Things angepasst ist. Die im Rahmen dieser Aktivität diskutierten Lösungsansätze orientieren sich ausdrücklich an einer REST-basierten Architektur (*Representational State Transfer*, [Fiel00]), einem heute gängigen Architekturmodell für Web-Anwendungen. Angelehnt daran kommunizieren Clients in ACE mit sogenannten Ressourcenservern, die in Ressourcen Daten von Interesse speichern. Darüber hinaus bestehen einige Unterschiede in den Lösungsansätzen. Abbildung 2 illustriert die relevanten Architekturmodelle, die im Folgenden näher erläutert werden.



(a) OAuth/UMA-Modell **(b)** DCAF-Modell

Abb. 2: Architekturmodelle für die Autorisierung

Tschofenig [Tsch15] beschreibt ein Profil, das OAuth auf das Internet of Things anwendbar machen soll. Das zugehörige Rollenmodell visualisiert Abbildung 2a: Ein Client agiert im Sinne des *Requesting Party* (RqP) genannten Nutzers. Um Zugriff auf eine geschützte Ressource zu erhalten, muss der Client gegenüber dem betreffenden Ressourcenserver ein sogenanntes *Access Token* vorweisen können, das speziell für diesen Zugriff von einem AS eingeholt worden ist. Der AS ist aus Sicht des Ressourcenservers eine vertrauenswürdige Instanz, die vom Besitzer des Smart Objects (*Resource Owner*, RO) mit Autorisierungsregeln für dessen Ressourcen konfiguriert worden ist. Anhand dieser Regeln entscheidet der AS, ob der Anfrage des Clients stattgegeben wird und stellt in diesem Fall das erforderliche Access Token aus.

Da die Umsetzung der Autorisierungsentscheidung im Ressourcenserver erfolgt, muss dieser anhand des präsentierten Access Tokens überprüfen können, ob es von einem dazu berechtigten AS ausgestellt worden ist. OAuth realisiert dies über eine kryptographische Bindung des Access Tokens an bestimmte Attribute des AS, beispielsweise ein X.509-Zertifikat [ITUT12, SaHo11].

[5] https://datatracker.ietf.org/wg/ace/charter/

Um die Berechtigung des AS zu prüfen, benötigt der Ressourcenserver zusätzliche Konfigurationsdaten, etwa das X.509-Zertifikat des AS. Weiterhin muss die zugehörige Konfiguration auf dem Smart Object geändert werden, wenn der AS gewechselt werden soll. Eine geeignete Schnittstellendefinition dafür beschreiben Tschofenig et al. [TMWE15] als Erweiterung des OAuth-Pofils *User-Managed Access* (UMA).

Zur Änderung der Rechtekonfiguration legt UMA eine eigene Schnittstelle fest, die *Protection API*. Die Autorisierung für Zugriffe auf die Protection API erfolgt wiederum über OAuth. Die Zuständigkeit eines Autorisierungsservers für einen Ressourcenserver manifestiert sich in der kryptographischen Bindung von Ressourcenserver, Autorisierungsserver und dem Besitzer der betreffenden Ressourcen zu einem *Protection API Token* (PAT). Eine Änderung des zuständigen Autorisierungsservers würde somit durch Ausstellung eines neuen PAT erreicht.

Das Delegated CoAP Authentication and Authorization Framework (DCAF, [GeBB15]) verknüpft Autorisierungstickets mit dynamisch erzeugtem Schlüsselmaterial für die gesicherte Kommunikation von Smart Objects. Die Berechtigung des *serverseitigen Autorisierungsservers* (SAS), eine Autorisierungsentscheidung im Namen des Ressourcenbesitzers (RO) zu treffen, ist in DCAF unmittelbar an ein kryptographisch erzeugtes Ticket geknüpft, aus dem ein symmetrischer Schlüssel für die gesicherte Kommunikation zwischen Client und Ressourcenserver abgeleitet wird. Für eine bessere Unterstützung des Clients kann optional ein zweiter Autorisierungsserver (*clientseitiger Autorisierungsserver*, CAS) eingesetzt werden. Dem Besitzer des Clients wird dadurch mehr Kontrolle über sein Smart Object gegeben.

Das ausgestellte Ticket ist so beschaffen, dass es eine gegenseitige Authentisierung von Client und Server ermöglicht. Die Konfiguration dieser beiden Komponenten beschränkt sich daher jeweils auf die Ausstattung mit einem kryptographischen Schlüssel ihres zugeordneten AS. DCAF ist dadurch in der Lage, Clients der Klasse 1 zu unterstützen, wie Abbildung 2b verdeutlicht.

Die obige Beschreibung der innerhalb der Arbeitsgruppe ACE diskutierten Ansätze zeigt, dass je nach Anwendungsgebiet und vorausgesetzter Leistungsfähigkeit der Smart Objects sehr unterschiedliche Technologien zum Tragen kommen. Sowohl das UMA-Profil für OAuth als auch DCAF bieten grundsätzlich eine Möglichkeit zum Ändern des zuständigen AS während unterschiedlicher Phasen im Lebenszyklus eines Smart Objects und erkennen den besonderen Schutzbedarf dieser Modifikation an. Eine eingehende Beschreibung der Schutzmaßnahmen, die ergriffen werden müssen, um den Wechsel des AS oder auch des Inhabers der Autorisierungsberechtigung abzusichern, steht jedoch noch aus. Eine mögliche Vorgehensweise mit DCAF beschreibt unser im IETF-Begleitdokument [Gerd15] spezifizierte Ansatz, der im folgenden Abschnitt näher beleuchtet wird.

5 Management von Autorisierungsberechtigungen

Wie bereits in Abschnitt 3 beschrieben, sind die Phasen im Lebenszyklus des Geräts, in denen sich der Besitzer des Smart Objects ändert, besonders zu schützen. In der technischen Umsetzung trifft der AS stellvertretend für den Besitzer die Autorisierungsentscheidungen für ein Smart Object. Entsprechend ist der Wechsel des AS speziell zu schützen.

Das Smart Object muss validieren können, ob Autorisierungsentscheidungen, also zum Beispiel Autorisierungstickets, von einer dazu berechtigten Instanz kommen, das heißt von seinem zuständigen AS. Dazu benötigt das Smart Object Schlüsselmaterial, das mit dem AS assoziiert ist.

Wir nennen dieses Schlüsselmaterial AS-Key. Der AS beglaubigt von ihm getroffene Autorisierungsentscheidungen mithilfe kryptographischer Verfahren, wobei sowohl symmetrische als auch asymmetrische Algorithmen verwendet werden können. Bei asymmetrischen Verfahren verwaltet AS den privaten Schlüssel zu dem AS-Key und signiert damit die Autorisierungstickets. Das Smart Object nutzt den öffentlichen Schlüssel zur Überprüfung der Signatur. Bei symmetrischen Verfahren teilen AS und Smart Object einen gemeinsamen AS-Key, der sowohl zum Beglaubigen als auch zum Validieren verwendet wird.

Da in REST-basierten Systemen Informationen in Ressourcen gespeichert werden, liegt es nahe, für das AS-Schlüsselmaterial ebenfalls eine Ressource vorzusehen. Diese kann durch die Autorisierungslösung ebenso geschützt werden wie andere Ressourcen. Die Zugriffsberechtigungen für die AS-Key-Ressourcen sind allerdings besonders sorgfältig zu definieren. Nur Subjekte, die im Besitz der Autorisierungsberechtigung sind, sollten Schreibrechte für diese Ressource bekommen. Wenn die Ressource einen symmetrischen Schlüssel enthält, sollte sie nicht lesbar sein, damit sie nicht von Unbefugten ausgelesen werden kann.

Im Folgenden werden die Maßnahmen zur Sicherung der Autorisierungsberechtigung im Lebenszyklus eines Smart Objects beschrieben, die für alle REST-basierten Autorisierungslösungen im Internet of Things nutzbar sind.

5.1 Autorisierungstickets

Wenn sich der AS für ein Smart Object ändert, muss die AS-Key-Ressource entsprechend geändert werden. Um dies zu erreichen, wird eine Bestätigung in Form eines Autorisierungstickets benötigt, das den Schreibzugriff auf die AS-Key-Ressource erlaubt. Es gibt drei Möglichkeiten, ein solches Ticket zu erhalten:

1. Der bisherige AS stellt das Ticket aus.
2. Ein fertiges, vorkonfiguriertes Ticket wird verwendet.
3. Eine Kopie des alten AS-Key (bei symmetrischen Schlüsseln), beziehungsweise des zugehörigen privaten Schlüssels (bei asymmetrischen Schlüsseln) wird verwendet, um das Ticket zu erstellen.

Üblicherweise erstellen Autorisierungsserver die Tickets. Entsprechend kann auch in dem Fall, wo sich der AS ändert, ein Ticket von dem bisherigen AS ausgestellt werden. Dazu muss dieser aber durch den neuen AS erreichbar sein und dazu konfiguriert, ein solches Ticket auszustellen. Das ist aber nicht immer der Fall.

Die Verwendung eines vorkonfigurierten Tickets kann Abhilfe schaffen, wenn der bisherige AS nicht erreichbar ist, zum Beispiel weil er außer Betrieb gesetzt wurde.

Schließlich ist es auch möglich, mithilfe des Schlüssels des bisherigen AS ein entsprechendes Ticket zu erstellen. Da dieser Schlüssel besonders geschützt werden muss, ist es allerdings nicht ratsam, ihn an andere zu übertragen.

Die folgenden Abschnitte beschreiben, wie Autorisierungstickets in den einzelnen Phasen des Lebenszyklus zur Übertragung von Autorisierungsberechtigungen verwendet werden.

5.2 Herstellung

In der Herstellungsphase werden Geräte typischerweise mit initialem Schlüsselmaterial ausgestattet. Dadurch wird später dem rechtmäßigen Besitzer ermöglicht, eine Vertrauensbeziehung

zu dem Gerät aufzubauen. Dem Hersteller stehen drei Provisionierungsoptionen zur Verfügung:

- Provisionierung mit Autorisierungsserver
- Provisionierung ohne Autorisierungsserver
- Keine Provisionierung

Wenn das Smart Object mit Schlüsselmaterial ausgestattet wird, kann der Hersteller entscheiden, dass er einen eigenen AS betreiben will. In diesem Fall wird die AS-Key-Ressource des Smart Objects mit dem entsprechenden AS-Key initialisiert. Dem zugehörigen AS wird die Autorisierungsberechtigung für das Smart Object zugeteilt, indem ihm das zugehörige Schlüsselmaterial bekannt gemacht wird. Bei der Verwendung von symmetrischen Schlüsseln sollte der Hersteller jedem Smart Object einen eigenen Schlüssel zuweisen, um Angriffen vorzubeugen. Der zukünftige Besitzer kann sich bei dem AS anmelden, um ein Ticket zur Übertragung der Autorisierungsberechtigung zu beantragen.

Die Provisionierung ohne AS erfordert ebenfalls die Initialisierung der AS-Ressource mit dem AS-Key. Da kein AS zur Verfügung steht, muss den zukünftigen Besitzern das Schlüsselmaterial für die Autorisierungsberechtigung auf andere Weise zugänglich gemacht werden, zum Beispiel indem es dem Produkt in Form eines QR-Codes beigelegt wird.

Falls keine Provisionierung stattfindet, wird die AS-Key-Ressource nicht initialisiert und muss ungeschützt bleiben, damit sie vom zukünftigen Besitzer gesetzt werden kann. In dem Fall kann sie ohne Ticket einfach beschrieben werden, um die Autorisierungsberechtigung zu vergeben. Da das Gerät in diesem Fall völlig ungeschützt gegen Angriffe ist, wird diese Vorgehensweise im Folgenden nicht weiter betrachtet.

5.3 Inbetriebnahme

Bei Inbetriebnahme geht das Smart Object logisch vom Besitz des Herstellers in den Besitz des Eigentümers über. Um die Kontrolle über das Gerät zu erlangen, muss die Autorisierungsberechtigung an den neuen Besitzer übergehen. Hierfür ist nur eine Maßnahme erforderlich: der Austausch von Schlüsselmaterial zwischen AS und Smart Object.

Bietet der Hersteller einen AS an, beantragt der neue AS dort ein entsprechendes Ticket, das es ihm erlaubt, die AS-Key-Ressource auf dem Smart Object zu ändern (siehe auch Abschnitt 5.1).

Mithilfe des Tickets kann dann die AS-Key-Ressource neu gesetzt werden. Dabei ist zu beachten, dass die Übertragung des neuen Schlüsselmaterials gesichert erfolgen muss: Die Integrität und Authentizität der Nachricht müssen sichergestellt sein. Bei symmetrischen Schlüsseln ist zusätzlich die Vertraulichkeit zu gewährleisten. Da der bisherige AS den neuen AS-Key nicht kennen sollte, sind spezielle Schutzmaßnahmen erforderlich. Durch das Setzen der AS-Key-Ressource wird der alte AS-Key überschrieben. Dadurch wird dem bisherigen AS die Möglichkeit entzogen, gültige Tickets auszustellen.

Falls das AS-Schlüsselmaterial dem Produkt beigelegt ist, wird es vom neuen Autorisierungsmanager dazu verwendet, ein Ticket zum Beschreiben der AS-Key-Ressource zu erstellen.

5.4 Betrieb

Während der Betriebsphase entscheidet der AS für das Smart Object, ob der Zugriff auf eine Ressource gestattet ist. Das Smart Object überprüft Autorisierungstickets mithilfe des AS-Keys.

5.5 Wartung

Von Zeit zu Zeit kann es nötig werden, den zuständigen AS für ein Smart Object auszutauschen, um Wartungsmaßnahmen durchführen zu können. Für einen sicheren Austausch sind die folgenden Maßnahmen erforderlich: Ausstellen eines Tickets für den neuen AS und das Löschen von Daten auf dem alten AS, damit dieser vorübergehend keine neuen Tickets mehr ausstellt.

Bevor der bisherige AS ausser Betrieb genommen wird, sollte von ihm ein Ticket erstellt werden, um die AS-Key-Ressource zu ändern und so die Autorisierungsberechtigung zu übertragen. Nach der Änderung der Ressource kann der bisherige AS keine gültigen neuen Tickets mehr ausstellen. Dennoch sollte von ihm das alte Schlüsselmaterial sowie eventuell gespeicherte Tickets gelöscht werden. Da sich an der Vertrauenswürdigkeit von Kommunikationspartnern in bestehenden Verbindungen, für die alte Tickets verwendet werden, nichts geändert hat, müssen alte Tickets nicht widerrufen werden und können weiterbestehen.

5.6 Außerbetriebnahme

Wenn Smart Objects außer Betrieb genommen werden, muss sichergestellt sein, dass ihnen alle Berechtigungen entzogen werden, damit ein Angreifer, der sie in seinen Besitz bringt, sie nicht als Einfallstor für seine Angriffe nutzen kann. Dazu sind die folgenden Maßnahmen erforderlich: Deregistrieren beim zuständigen AS, Entwerten von alten Tickets und das Löschen von Daten auf dem Smart Object sowie gegebenenfalls das Zurücksetzen der AS-Key-Ressource.

Zum Entzug der Berechtigungen müssen die Smart Objects beim zuständigen AS deregistriert werden, so dass dieser zukünftig keine weiteren Tickets mehr für sie ausstellt. Andere Geräte im Netz müssen darüber informiert werden, dass ein Smart Object nicht mehr autorisiert ist, indem bereits ausgestellte Tickets für ungültig erklärt werden. Auf den ausgemusterten Smart Objects sind Sitzungsschlüssel und gespeicherte Tickets zu löschen. Ausgemusterten Geräten wird so der Zugriff auf geschützte Ressourcen verwehrt, und es wird verhindert, dass fälschlich auf eine nicht mehr vertrauenswürdige Ressource zugegriffen wird. Der AS-Key darf nicht gelöscht werden, wenn er dafür verwendet werden soll, einen neuen AS-Key zu setzen (siehe auch Abschnitt 5.7). Da dieses Schlüsselmaterial nur dazu verwendet wird, mit dem AS zu kommunizieren, hat ein Angreifer nach der Deregistrierung keine Möglichkeit mehr, es zu missbrauchen.

5.7 Übergabe

In der Übergabephase wechselt das Smart Object den Besitzer und wird in ein neues Netz eingeführt. In diesem Fall muss die Autorisierungsberechtigung übertragen und dem alten Besitzer entzogen werden. Dazu ist eine Kombination der Maßnahmen für die Außerbetriebnahme und die Inbetriebnahme erforderlich: Im alten Netz muss die Deregistrierung bei dem alten AS, das Entwerten alter Tickets und das Löschen von Daten auf dem Smart Object erfolgen. Im neuen Netz müssen der neue AS und das Smart Object miteinander bekannt gemacht werden.

Um die Autorisierungsberechtigung auf einen neuen AS zu übertragen, stehen die in Abschnitt 5.2 beschriebenen Provisionierungsmöglichkeiten zur Verfügung: der neue AS kann bei dem alten ein Ticket beantragen, der alte AS stellt ein Ticket zum Setzen der AS-Key-Ressource aus oder der Schutz der AS-Ressource wird aufgehoben.

In dem alten Netz muss eine Außerbetriebnahme erfolgen wie in Abschnitt 5.6 beschrieben. Die Inbetriebnahme im neuen Netz erfolgt gemäß den Erläuterungen in Abschnitt 5.3.

Wenn eine große Anzahl von Geräten übergeben wird, zum Beispiel weil ein Haus verkauft wird, das mit einer Heimautomatisierungslösung ausgestattet ist, kann es sinnvoll sein, den AS mit zu übergeben. In diesem Fall müssen die Login-Daten geändert werden, mit denen sich der Nutzer auf dem AS anmeldet. Beziehungen zwischen dem AS und allen Geräten, für die er nicht länger zuständig ist, müssen aufgelöst werden wie in Abschnitt 5.5 beschrieben.

5.8 Fallback-Mechanismus

Die AS-Key-Ressource beinhaltet das Schlüsselmaterial für die Autorisierungsberechtigung. Wer sie überschreiben kann, kann gültige Autorisierungstickets ausstellen. Der Schutz dieser Ressource ist daher essentiell für die Sicherheit des Smart Objects.

Dieser Schutz kann aber auch zum Problem werden, wenn das Schlüsselmaterial zum Erstellen von Tickets nicht verfügbar ist. Das kann beispielsweise der Fall sein, wenn der Vorbesitzer sich weigert, den Schlüssel zu übergeben, wie etwa bei Zwangsversteigerungen, oder auch wenn ein AS aufgrund eines Defekts ausfällt. In solchen Fällen wird das Smart Object unbrauchbar, wenn die AS-Key-Ressource nicht gesetzt werden kann, ohne den AS-Key zu kennen.

Um dieses Problem zu umgehen, wird ein Rücksetz-Mechanismus benötigt, der berechtigten Nutzern das Überschreiben der AS-Key-Ressource ermöglicht, ohne Angreifer in die Lage zu versetzen, diesen Mechanismus für sich auszunutzen. Wie ein Fallback-Mechanismus aussehen muss, hängt stark vom Einsatzgebiet ab. Häufig wird dafür ein Reset-Knopf auf dem Gerät vorgesehen, der betätigt werden kann, um das Gerät auf Werkseinstellungen zurückzusetzen. Die AS-Key-Ressource kann dann überschrieben werden wie in Abschnitt 5.3 beschrieben.

Wenn das Smart Object an einem Ort eingesetzt wird, an dem es für Außenstehende zugreifbar ist, ist dieser Mechanismus aber problematisch. Er könnte von Angreifern dazu missbraucht werden, den Betrieb zu stören oder sogar das Gerät zu übernehmen. In einigen Fällen kann es daher sogar sinnvoll sein, auf einen Rücksetz-Mechanismus zu verzichten und das Gerät auszutauschen, wenn der Schlüssel verloren geht.

5.9 Implementierungsbeispiel DCAF

Dieser Abschnitt beschreibt die Umsetzung der erforderlichen Maßnahmen mit DCAF.

Bei der Verwaltung der Autorisierungsberechtigungen nimmt das Smart Object immer die Rolle des Ressourcenservers ein, auch wenn es sonst einen Client darstellt. Der zukünftige AS nimmt die Rolle des Clients ein. Es werden zwei Ressourcen benötigt: die AS-Key-Ressource zum Speichern des AS-Key sowie eine AS-URI-Ressource, die den URI des zuständigen AS enthält, bei dem Tickets zur Änderung der Ressource beantragt werden können.

Die AS-Key-Ressource wird mit einem symmetrischen Schlüssel initialisiert, typischerweise durch den Hersteller. Bei einer Provisionierung mit Autorisierungsserver wird zudem der URI, bei dem Ticket-Requests gestellt werden können, in AS-URI eingetragen. Sobald DCAF aktiviert ist, können Ressourcen nicht mehr ohne Ticket geändert werden.

Ein DCAF-Ticket zum Ändern der AS-Key-Ressource enthält deren URI, das benötigte Zugriffsrecht, in diesem Fall PUT, sowie einen Zeitstempel und eine Gültigkeitsdauer für das Ticket. Um ein Ticket für den Zugriff auf die AS-Key-Ressource zu bekommen, sendet der neue AS eine Anfrage für den AS-Key-URI an den bisherigen AS. Ist dieser nicht bekannt, kann er ermittelt werden, indem ein unautorisiertes Request für die AS-Key-Ressource an das Smart Object gesendet wird. Es weist die Anfrage zurück, sendet dabei aber den AS-URI mit.

Mit dem Ticket kann der zukünftige AS an das Smart Object die Anfrage stellen, um die Ressource zu ändern. Ein Smart Object, das ein Ticket bekommt, validiert es mithilfe des AS-Key in der AS-Key-Ressource. Damit der ehemalige AS den neuen Schlüssel nicht mitlesen kann, muss ein Schlüsselaushandlungsverfahren wie Diffie-Hellman [DiHe76] zwischen dem neuen AS und dem Smart Object verwendet werden.

Wenn der AS eine Anfrage zum Übertragen der Autorisierungsberechtigung bekommt, darf er danach keine neuen Tickets mehr ausstellen, weil er nicht dafür bürgen kann, dass er für dieses Smart Object noch zuständig ist. Bereits ausgestellte Tickets für das Smart Object müssen von ihm widerrufen werden.

DCAF definiert keinen Mechanismus zum Widerrufen von Tickets und müsste entsprechend erweitert werden. Dazu müsste auf dem Smart Object eine Revocation-Ressource angelegt werden, die angesprochen wird, um das Smart Object über abgelaufene Tickets zu informieren. Das Smart Object würde dann bei eingehenden Requests prüfen, ob deren Tickets noch gültig sind.

6 Fazit und Ausblick

Heute gängige Sicherheitslösungen wurden für Web-Umgebungen und leistungsstärkere Geräte entwickelt. Diese Lösungen sind aufgrund der spezialisierten Einsatzgebiete und der Einschränkungen von Smart Objects für diese nicht ohne weiteres anwendbar. Dennoch müssen Smart Objects in der Lage sein, grundlegende Sicherheitsaufgaben zu erfüllen, um die Sicherheitsziele ihrer Anwender zu erreichen.

Der vorliegende Beitrag diskutiert die Rolle von Autorisierung im Internet of Things vor dem Hintergrund des Lebenszyklus von Smart Objects. Eine Analyse von Vorschlägen aus der Arbeitsgruppe ACE der IETF zeigt, dass eine effektive Umsetzung des Nutzerwillens eine enge Bindung zwischen Smart Object und Autorisierungsserver voraussetzt. Dazu werden beide Komponenten mit dem kryptographischen Material ausgestattet, das bei einem Wechsel des Autorisierungsservers geändert werden muss.

Zwar sind die notwendigen Mechanismen für die Änderung der Konfiguration in den Protokollentwürfen vorgesehen, aber es fehlt eine detaillierte Beschreibung der Maßnahmen, die ergriffen werden müssen, um den Schutz der Autorisierungsinformationen im gesamten Lebenszyklus eines Gerätes sicherzustellen. In diesem Beitrag wird das Management von Autorisierungen im Lebenszyklus eines Smart Objects untersucht und gezeigt, wie sich darauf aufbauend eine REST-basierte Autorisierungslösung für das Internet of Things verwirklichen lässt. Die Ausgestaltung mit REST-basierten Lösungen wie DCAF ermöglicht einen automatisierten Austausch des Autorisierungsservers, was insbesondere bei einer großen Anzahl von zu verwaltenden Smart Objects ein Vorteil ist.

Literatur

[BoEK14] C. Bormann, M. Ersue, A. Keranen: Terminology for Constrained-Node Networks. RFC 7228 (2014), Internet Request for Comments.

[DiHe76] W. Diffie, M. Hellman: New directions in cryptography. In: *IEEE Transactions on Information Theory*, 22, 6 (1976), 644–654.

[Fiel00] R. T. Fielding: Architectural Styles and the Design of Network-based Software Architectures. Dissertation, University of California, Irvine (2000).

[GeBB15] S. Gerdes, O. Bergmann, C. Bormann: Delegated CoAP Authentication and Authorization Framework (DCAF). draft-gerdes-ace-dcaf-authorize-02 (2015), Internet-Draft (Work in Progress).

[Gerd15] S. Gerdes: Managing the Authorization to Authorize in the Lifecycle of a Constrained Device. draft-gerdes-ace-a2a-00 (2015), Internet-Draft (Work in Progress).

[Hard12] D. Hardt: The OAuth 2.0 Authorization Framework. RFC 6749 (2012), Internet Request for Comments.

[HSRV+14] R. Hummen, H. Shafagh, S. Raza, T. Voigt, K. Wehrle: Delegation-based Authentication and Authorization for the IP-based Internet of Things. *In: Proc. of IEEE SECON* (2014).

[HWZH+13] R. Hummen, H. Wirtz, J. Ziegeldorf, J. Hiller, K. Wehrle: Tailoring End-to-End IP Security Protocols to the Internet of Things. *In: Proc. of IEEE ICNP* (2013).

[ITUT12] ITU-T: X.509: Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks. Recommendation (2012).

[Moor65] G. E. Moore: Cramming more components onto integrated circuits. In: *Electronics*, 38, 8 (1965).

[SaHo11] P. Saint-Andre, J. Hodges: Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS). RFC 6125 (2011), Internet Request for Comments.

[SBJM+14] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, C. Mortimore: OpenID Connect Core 1.0. Technical Specification (2014).

[SGSM+15] L. Seitz, S. Gerdes, G. Selander, M. Mani, S. Kumar: ACE use cases. draft-ietf-ace-usecases-03 (2015), Internet-Draft (Work in Progress).

[ShHB14] Z. Shelby, K. Hartke, C. Bormann: The Constrained Application Protocol (CoAP). RFC 7252 (2014), Internet Request for Comments.

[TMWE15] H. Tschofenig, E. Maler, E. Wahlstroem, S. Erdtman: Authentication and Authorization for Constrained Environments Using OAuth and UMA. draft-maler-ace-oauth-uma-00 (2015), Internet-Draft (Work in Progress).

[Tsch15] H. Tschofenig: The OAuth 2.0 Internet of Things (IoT) Client Credentials Grant. draft-tschofenig-ace-oauth-iot-01 (2015), Internet-Draft (Work in Progress).

[TsFo15] H. Tschofenig, T. Fossati: A TLS/DTLS Profile for the Internet of Things. draft-ietf-dice-profile-12 (2015), Internet-Draft (Work in Progress).