DEZSYS AUSARBEITUNG

Philipp Adler

***Abstract*—Die Technologie breitet sich in unserer modernen Welt immer aus. Digitale Devices sind Gegenstände die uns im Alltag begleiten und unser Leben einfacher gestalten. Wir benutzen täglich unser Smartphone, schauen Videos über dem Laptop oder betreiben schon fortgeschrittenen Sport mit Sensoren. All diese Geräte waren vor Jahren kaum vorstellbar und auch noch ziemlich teuer zu leisten. In den letzten Jahren ist der Preise für diese sogenannten Gadgets rapide gefallen. In Zukunft sollen uns elektronische Geräte nicht nur in unserer Freizeit, sondern auch zuhause, in einem "Smart Home, helfen. Wie der Name schon sagt, sind "Smart Homes" automatisierte Häuse, die Routineaufgaben des Menschen erledigen sollen. Mithilfe des IPv6 Standards wird es möglich, dass jedes Gerät seine eigene Adresse hat und somit in der Lage ist, mit anderen Microsensoren zu kommunizieren bzw. Daten auszutauschen, um die Anforderungen und Wünsche des Menschens zu erfüllen.**

I. CLOUD COMPUTING AND INTERNET OF THINGS

Das Thema Cloud Computing hat in den letzten Jahren stark an Bedeutung gewonnen. Es erlaubt Endanwendern ihre Daten überall abzurufen. Die Daten werden in eine Wolke, für den User, ein nicht einsehbares Netzwerk, gespeichert. Es bietet Firmen die Möglichkeit, flexibler und billiger Daten zu speichern und dem Endkunden Service anbieten. Es ist schwier, dem Server, in kurzer Zeit Rechenleistung zuzufügen. Bei einer Cloud ist dies Standard.

Bei einem Smart Home fungiert ein Router als Middleware zwischen den Sensoren. Die Middleware bietet den Vorteil, dass verschieden Technologien genutzt werden können. Es macht nur Sinn Eine zu verwenden, wenn die einzelnen Geräte keine IP-Adresse haben. Andernfalls benötigt es keine Middleware, da sich die Cloud hingegen direkt verbinden kann. Mittels IPv6 können Geräte untereinander kommunizieren. So hat man auch als Anwender eines Smart Homes die Möglichkeit, Remote, also mit Fernzugriff, die einzelnen Geräte zu steuern, automatisierte Prozesse zu konfigurieren und auch diese mittels Logs zu überwachen. Dank der Hochverfügbarkeit der Cloud, hat man jederzeit Zugriff auf sein Zuhause.
Neben den tollen Features, gibt es zusätzlich, je nach Anwendungszweck, drei Cloudparadigmen, auf die im kommenden Kapitel noch näher eingegangen wird.[1]

A. LB App-Ebene

Der Load Balancer, auch Content Switch auf Applikationsebene, bezieht sich auf die Verteilung von Contentanfragen. Geräte, die Layer 7 Load Balancing betreiben können, heißen Application Delivery Controller(ADC). Der Load Balancer ist für die Außenwelt ein virtueller Server, der Anfragen annimmt und diese aufgrund von Applikationsrichtlinien an die Server verteilt. Diese Server wurde so optimiert, dass sie nur für einen Content zuständig sind. So wird die bestmögliche Performance erzielt. Um gegen Ausfälle vorzubeugen, befinden sich mehrere Server, die sich auf dasselbe konzentrieren, in einem Pool. In der Cloud, kann Parallelisierung mit einer Load balancing Anwendung oder mittels Content Switching betrieben werden, indem die eingehenden Anfragen an die virtuellen Maschinen verteilt werden.[2]

B. The Internet of Things

IoT verbindet die Informationen von Geräten, wie Sensoren oder von Aktoren. IoT selbst bietet eine Reihe an Technologien für das Kontrollieren und Erreichen von Komponenten an. Zu Beginn gab es die RFID Etikette, welche Produktinformationen enthält und sich weiter in Richtung des automatisierten Nachrichtenaustausches mittels Wireless Sensor and Actuator Networks, kurz WSANs, entwickelte. Ein WSAN besteht aus mehreren wireless Knoten, genannt *Sinks*, hinter denen sich entweder ein Sensor oder Akteur befindet. Die Informationen werden von Knoten zu Knoten weitergeleitet, bis das Ziel erreicht ist. Dabei ist zu erwähnen, das Netzwerk mitlernt und so den kürzersten Weg findet.
Weiters können Daten auch mittels Machine-to-Machine(M2M) kommuniziert werden. Es besteht die Möglichkeit mit Kabel oder Funk anzusetzen. Dabei gibt es drei Komponenten. Die Endpunkte, die

Informationen untereinander oder mit der zentralen Leistelle austauschen. Es ermöglicht mit Funkt Daten zu übertragen und auch remote die Geräte zu steuern.[1]

1) 6LoWPAN: IPv6 over Low power Wireless Personal Area Network ist ein Protokoll, welches den Geräten erlaubt IPv6 zu benutzen, währrend sie über WLAN über UDP kommunizieren. Es bietet die Möglichkeit drahtlose PANs in ein bestehendes Netz zu integrieren. Sozusagen wird ein drahtloses Heimnetzwerk, bestehend aus embedded Devices, mit dem Internet verbunden. Für die sichere M2M-Kommunikation im Netzwerk können IPsec oder Zertifikate verwendet werden. Der Router 6LBR verbindet ein Senornetzwerk(WSN) mit einem 6LoWPAN, basierend auf Ethernet IPv6. Der Router verwaltet das Smart Home, während sich auch um den Datenaustausch zwischen dem Netzwerk kümmert. Sozusagen ist es für die IP-Kommunikation der Sensoren zuständig.
Um die Funktionsweise von 6LoWPAN zu verstehen, muss einen der Unterschied von IPv4 und IPv6 bekannt sein. IPv4 bildet das heutige Internet, ist aber aufgrund des geringen Adressraumes von IPv6 ersetzt worden. IPv6 bieten eine Adressraum von 2^{128}. Auch der IPv6-Header ist für das Verständnis des 6LoWPAN Verfahren relevant. Der Header besteht aus acht Feldern. Diese sind Version, im Falle von IPv6 Version 6, Traffic Class und Flow Label, für den Umgang der Pakete, Payload Length, Länge der Nutzdaten, Next Header, gibt das verwendete Protokoll an, Hob Limit, maximale Anzahl an Hops und die Source- und Zieladresse.
Neben dem Adressraum bietet IPv6 den Vorteil, dass es gestattet den Knoten automatisch zu konfigurieren. Der Host kommuniziert über das Neighbor Discovery Protocol(NDP) und erhält automatisch eine Adresse. Der Host sendet eine NS(Neighbor Soliciatation)-Nachricht an alle Teilnehmer im Netzwerk, um zu erfahren, ob die generierte IPv6 Adresse schon vergeben ist. Kommt nach einer bestimmten Zeit keine NA(Neighbor Advertisement)-Nachricht zurück, kann er davon ausgehen, dass die Adresse nicht belegt ist. Um nun das korrekte Netzwerk-Präfix zu erhalten sendet der Host an den Router eine NS-Nachricht, der mit einer Advertisment Nachricht, das korrekte Präfix zurückschickt.
Der IPv6-Header hat eine maximale Größe von 127 Bytes, wofür 25 Byte für den MAC Header, 40 Bytes für den IPv6 Header und nur 62 Byte für die eigentlichen Daten übrig bleibt. Kommt dazu noch eine Verschlüsselung oder andere Tools, verringert sich der Platz für die eigentlichen Daten noch mehr.
Ein Problem stellt die MTU, Maximum Transmission Unit, dar. Diese besagt, dass kleiner als 1280 Byte übertragen werden können, was für IPv6 sehr wenig ist.
Hier kommt 6LoWPAN ins Spiel, es vereint die Header-Komprimierung, Fragmentierung, Routing und Autokonfiguration.[1], [3], [4]

Routing unter 6LoWPAN

Mittels Routing wird das Paket über mehrere Knoten an die Zieladresse geschickt. 6LoWPAN bietet zwei Verfahren, Mesh-Under, basierend auf MAC-Adressen und Route-Over, werden die Pakte über IP übertragen. Beim Mesh-Undo-Verfahren werden die Fragmente bis zum Zielknoten weitergeleitet und dort zusammengestellt. Beim Route-Over-Verfahren hingegen werden die Fragemente nur bis zum nächsten Knoten geschickt, zusammengesetzt, ausgewertet und dann erst zum nächsten Knoten geschickt. Vorteil von Route-Over ist, dass alle Pakete sicher ankommen. Wenn bei Mesh aber die Pakete verloren gehen, muss das ganze Paket neu angefordert werden. Hingegen müssen die Pakete nicht abgespeichert werden, aber es kommt schnell zu hohen Netzwerkverkehr.
Obwohl Fragmentierung toll ist, ist es besser diese zu vermeiden. Dadurch steigt nur der Verwaltungsaufwand. Ohne senkt man die Netzwerklast und sorft für eine länge Gerätelebensdauer.[4]

2) CoAP: CoAP, ausgelegt für kleine, stromsparende Geräte, ist bestens geeignet für IoT. Applikationen z.B. reden über das CoAP Protokoll. In einem Smart Home, wo sich die Sensoren und Geräte in einem 6LoWPAN befinden werden die ausgehenden XML basierenden Nachrichten an die Cloud mittels CoAP übertragen. Da CoAP alleine nicht sicher ist, baut meine eine Zwischenschicht, DTLS, ein. Im Gegensatz zu HTTP konzentriert sich es leichtgewichtiger und effizienter zu sein. Da es wie HTTP ein RESTful Protokoll ist, bietet es GET, PUT, POST und OBSERVE, zur Datendarstellung. Basierend auf UDP werden die Daten untereinander ausgetauscht. CoAP ist ein Webprotokoll, welches für eine M2M-Verbindung ausgelegt ist. Mittels der URI kann auf die Geräte zugegriffen werden. CoAP wurde so designt, dass anstelle eines komplexen Transportstapel, wie HTTP, UDP verwendet und eine fixe Headergröße hat. Durch die kleine Größe, ist keine Fragmentierung notwendig. Für die Sicherheit verwendet CoAP DTLS.[1]
CoAP und HTTP sind network-oriented Protokolle, die

beide ähnliche Funktionen anbieten. CoAP hingegen bietet aber nocht Features wie wenig Overhead und Bandbreite oder Multicast, Gruppenkommunikation, an. HTTP basiert auf TCP und verwendet eine P2P-Kommunikation, die zu komplex für kleine Geräte ist. CoAP verwendet UDP, also es nicht so komplex und besstens für IoT geeignet. Als Ausgleich gegen die Unzuverlässigkeit von UDP, definiert CoAP einen Übertragungsmechanismus und bietet mit Ressource Discovery-Mechanismus mit Ressourcenbeschreibung.[4]

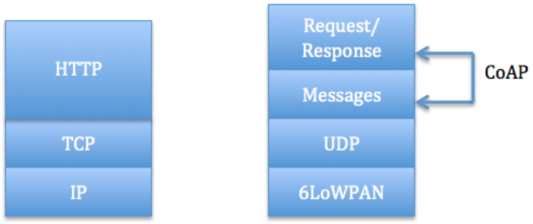

Fig. 1. HTTP und CoAP Protokollstapel [4]

CoAP Model

Beim Message Layer gibt es 4 Nachrichtentypen. Diese sind CON, NON, ACK, RST, wobei jeder eine ID enthält.

Zuverlässiger Nachrichtenaustausch: Beim Senden einer CON-Nachricht an den Server, wartet der Client auf ein ACK mit der selben ID. Kommt es zu keiner Antwort oder zu einem Timeout, wird die Nachricht einfach nochmal versendet.

Unverlässiger Nachrichtenaustausch: Beim senden einer NON-Nachricht, muss der Server nicht antworten. Sollte es aber zu einem Fehler bei der Verarbeitung gekommen sein, liefert der Server RST(Reset).

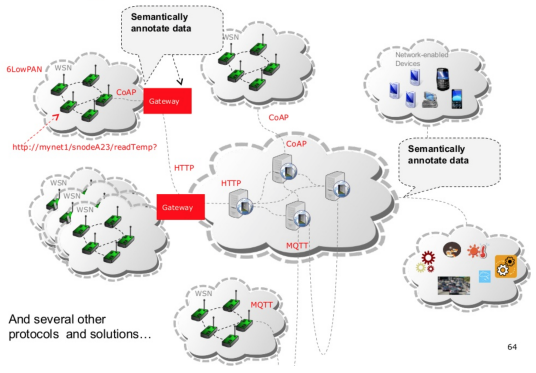

Fig. 2. Smart Home Architektur [5]

3) MQTT: Neben den Internetprotokoll HTTP und CoAP, etabliert sich das leichtgewichtige MQTT für das IoT. Es ist schlank und einfach zu inplementieren. Es ähnelt der JMS Architektur und implementiert ein Publish/Subscribe Pattern. MQTT bietet viele Übertragungsformate und einen minimalen Protokolloverhead. Dank Session-Aware, müssen die Metadaten nur einmal übertragen werden.

Nachrichten von den Publisher wird an den Broker in einen Topic gesendet. Diese Nachricht empfängt jeder Subscriber, nach dem Pushverfahren, der dieses Topic abboniert hat.

Zu Beginn verbindet sich der Client mit ein eindeutigen ID am Broker. Für Sicherheitszwecke kann der Broker so eingestellt werden, dass nur Client mit gültigen Anmeldeinformationen zugelassen werden. Außerdem gibt auch sogenannte QualityofService (QoS), die bestimmen wie wichtig es ist, dass die Nachricht auf der anderen Seite ankommt.

Nachteil von MQTT ist, dass die Nachricht als Klartext versendet wird. Es besteht die Möglichkeit den Broker mit einem SSL/TLS-Modul auszurüsten.[6]

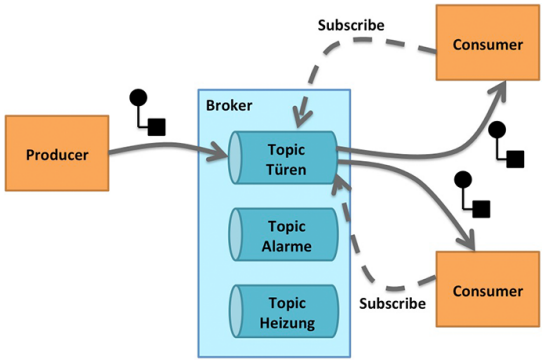

Fig. 3. MQTT Ablauf [6]

4) IoT Cloud Service Provider SicsthSense: SicsthSense ist ein Service für das Monitoring für automatisierte Geräte in einem Smart Home. Die Sensordaten können über die Verwendung einer API auf einer Webseite angezeigt werden. Die Cloud holt sich mittels HTTP und CoAP die Informationen von den Geräten.[1]

5) IoT Applocation Protocol oBIX: oBIX ist ein XML Standard für IoT Umgebungen, wo jedes Gerät durch eine eindeutige URI erreichbar ist. Hinter dieser URI befindet sich ein Objekt, welches die Informationen in Attributen, in den sogennannten ”facets”, speichert. Durch oBIX definiert, somit ein Standard Web-Service Protokoll, ermöglich die Kommunikation von verschiedenen Systemen. Die objektorientierten XML-Dateien werden über HTTP und SOAP übertragen. Dadurch ist es möglich die

Akteure mittels M2M zu steuern.
Aus Java Sicht aus, kann man sich das XML-File wie ein Interface vorstellen, welches nur die Struktur enthält, also Name und Datentyp der Variable, die von den konkreten Implementierungen umgesetzt werden müssen.[1]

C. Cloud Computing Layer & Frameworks

Ein Cloudsystem bietet drei Paradigmen, IaaS, Paas und Saas, an.
Infrastructure as a Service bietet die Möglichkeit seine eigenen Images oder bereits vordefinierte Images eines virtuelles Devices hochzuladen. Außerdem steht es dem Anwender frei, jede Technologie oder Tools auf die virtuelle Maschine zu installieren. Vorteil gegenüber PaaS ist, dass es mehr Möglichkeiten gibt, jedoch ist man für die Konfiguration selbst verantwortlich.
Bei Platform as a Service hat der Entwickler keine Möglichkeit, das Umfeld zu verändern. Er kann nur mittels der API auf die Ressourcen zuzugreifen. Als Vorteil kann der geringe Aufwand gesehen, da man nur für die Funktionalität der Applikation zuständig ist.
Software as a Service, bietet den User mittels einem Webinterface Zugriff auf die Software.[1]

1) PaaS Provider Appscale: Appscale ist ein Open-Source PaaS, welches die Programmiersprachen Python und Java unterstützt. Um Applikation lauffähig zu machen, ist Appscale kompatibel mit der Google App Engine(GAE) API. GAE Web Applikations können entweder auf einer Cloud oder einem Cluster bzw. auf verschiedenen Infrastrukturen deployed werden.[1]

2) Vergleich: PaaS ist gut, wenn man schnell eine Webapplikationen erstellen möchte, aber ungeeignet für ein Smart Home. Es fehlt z.B. die Funktionalität, dass sich Geräte mittels CoAP verbinden können und es besteht keine Möglichkeit einer Geräteüberwachung. Eine Lösung wäre, wenn die PaaS Anbieter eine geeignete IoT API anbieten.
Fertige SaaS-Systeme sind für Smart Homes besser geeignet, aber es werden nur für spezielle Services angeboten.[1]

3) PaaS Home Konzept: Wie schon erwähnt mangelt es bei vielen PaaS Services an den Kommunikationsprotokollen wie CoAP. Mit Appscale als Ausgangspunkt und seiner bestehenden API, welche sich auf oBIX-Format konzentriert, soll so erweitern werden, dass es für ein Smart Home geeignet ist. Die Geräte werden mit dem IoTSyS gateway, welches als Middleware fungiert, verbunden. Die Daten werden in den XML-Standard oBIX umgewandelt und mittels CoAP, geschützt mit DTLS, an die Cloud geschickt. Anfragen von Clients an die Cloud werden an das Gateway weitergeleitet, welches entweder die Geräte mit einer M2M-Kommunikation aktiviert, die Daten abfragt oder modifiziert und diese wieder an die Cloud zurückschickt. Mithilfe der Appscale Application ist es möglich, immer und überall die Daten über HTTP abzufragen.[1]

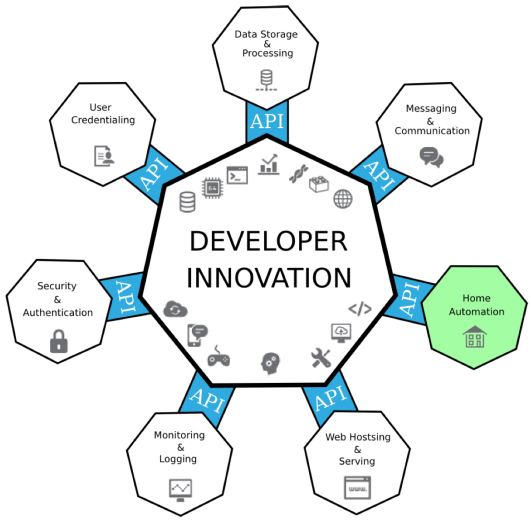

Fig. 4. Erweiterte Appscale API [1]

II. Automatisierung, Regelung und Steuerung

Aufgrund des fehlendem Interesse und dem Mangel an Wissen zu dem Thema Automatisierung, Regelung und Steuerung wird dieses Kapitel ausgelassen.

III. Security, Safety, Availability

In einem Smart Home spielen Faktoren wie Sicherheit und Hochverfügbarkeit eine wichtige Rolle. Wie kann ich sicherstellen, dass mein Zuhause von Hackern angegriffen wird und ein Zugriff von außen nicht möglich ist. Wie verteile ich am Besten meine Daten global? Wie sieht mein Backupkonzept aus, um sicherzustellen, dass keine Daten verloren gehen und welche Konzepte und Systeme eignen sich dafür? Diese Fragen werden im nachfolgendem Kapitel beantwortet.

A. Replikation/Allokation & Fragmentierung

Unter dem Begriff verteilte Datenbanksysteme versteht man, die Aufteilung der Daten. Mit der richtigen Verteilung kann eine Leistungssteigerung zustande kommen. Kommunikationaufwand, Lastenaufteilung und Hochverfügbarkeit, können mithilfe der Replikate, verbessert werden. Replikate können auch als Backup dienen. Bei einem Smart Home, wo so eine große Menge an Daten im Sekundentakt ankommt, ist es wichtig diese strategisch zu verteilen. Storj.io verwendet diese Konzept, indem die Daten in der Cloud verschlüsselt gespeichert werden. Die werden repliziert, in mehrere Junks aufgeteilt und auf verschiedene Standorte aufgeteilt, da der Speicher von irgendwelchen Nutzern zur Verfügung gestellt wird. Auch XDCR ist ein System, welche die aktiven Daten in einen anderen Cluster abspeichert bzw. repliziert.
Es gibt zwei Aufteilungsvarianten:

- Bottom-Up-Ansatz, mehrere Datenbanken werden in ein gemeinsames, konzeptionelles Schema zusammengefasst.
- Top-Down-Ansatz, hier wird das gemeinsame Schema, auf mehrere logische aufgeteilt.

1) Fragmentierung: Bei der Fragmentierung muss die Vollständigkeit, jedes Attribut muss in der gloablen Relation einmal vertreten sein, Rekonstruierbarkeit, die gespalteten Fragmente können wieder zusammengeführt werden und zum auf die Disjunktheit, Fragmente überlappen sich nicht, geachtet werden.
Die häufig verwendeste Fragmentierung ist die horizontale, die zeilenweise erfolgt. Die vertikale erfolgt spaltenweise. Für die Rekonstruierbarkeit müssen die einzelnen Fragmente gejoint werden, wobei jedes Fragment den Primärschlüssel enthalten muss. Da Fragmente selbst Relationen darstellen, können Fragmentierungsarten kombinert werden, auch hybride Fragmentierung genannt. [7]

2) Allokation und Replikation: Die Allokation ordnet die erstellten Fragmente den einzelnen Rechnern zu. Dabei können einzelne Fragmente repliziert werden. Man unterscheidet zwischen voller und partieller Replikation. Bei der Vollen wird die globale Relation in allen Rechnern vollständig gespeichert, was die Kommunikationsvorgänge verringert. Leseoperationen müssen nur mehr lokal ausgeführt werden. Außerdem führt eine volle Replikation zu einer Steigerung der Verfügbarkeit und zum Load Balancing. Nachteil ist, dass Änderungsvorgänge auf allen Replikaten übernommen werden müssen. Bei partiellen Replikation, die sich mit einer schwächeren Konsistenz zufriedenstellt, sind Teilmengen des Datenbestands redundant auf mehrere Rechnern. Es werden nicht alle Daten gespeichert. Bei keiner Replikation werden Fragmente nur auf einem Standort gespeichert.[7]

B. Backup

Viele IoT erstellen Daten in sekundentakt, die gebackupt werden müssen. Es benötigt eine Strategie, die unwichtigen Daten auszufiltern, währrend wichtige Daten gespeichert werden. Wichtig ist die Konfiguration der Geräte, sowie die Gerätefirmware zu speichern und zu beschützen.[8]

1) Network attached storage: NAS ist ein Speicher, der direkt mit dem Netzwerk verbunden ist. Die Redundanz wird in Form eines RAID-Systems angeboten. Die Daten werden auf mehreren Festplatten abgespeichert, sodass die Daten aus dem Netzwerk bezogen werden können.[8]

2) Cloud Storage: Das Online-Backup in einer Cloud, ist besser als die Daten auf einer externen Festplatte zu speichern. Bei Brand oder Naturkatastrophen stehen die Daten weiterhin in der Cloud zur Verfügung. Festplatte haben begrenzte Speicher, die ziemlich teuer sind. In einer Cloud hingegen, kann der Speicher immer erweitert werden und ist kostengünstiger.[8]

3) Private Cloud: Die Daten in der privaten Cloud stehen immer zur Verfügung, sind konsistent und vollständig. Es kann sogar auf frühere Versionen zugegriffen werden. Gespeicherte Daten sind überall aufrufbar und sind von entfernten Zugriffen geschützt. [8]

C. Confidentiality, Integrity & Availability (CIA)

Das CIA Dreieck, übersetzt Vertrauchlichkeit, Integrität und Verfügbarkeit, sind Sicherheitseigenschaften, die umbedingt eingehalten werden müssen, um sich vor Bedrohungen zu schützen. Z.b. IPsec unterstütz diese Sicherheitseigenschaften, um eine sichere Kommunikation zwischen den Sensoren zu gewährleisten.
Vertraulichkeit stellen Regeln dar, die den Datenzugriff von Informationen limitieren und die sichere Übertragung, mittels Verschlüsselung, gewährleisten. Ein Beispiel für Vertraulichkeit wäre die Zwei-Faktor-Authentifizierung, also die Kombination von einem Passwort und z.B. einem

Biometrischen Muster wie einem Fingerabdruck. Außerdem ist ein weiterer wichtiger Aspekt, dass die Daten nur in limitierter Anzahl gespeichert und gesendet werden. Im Falle Smart Home, kann nur der Kommunikationsendpoint die gesendeten Daten lesen.
Integrität stellt die Gewissheit dar, das die Daten vertrauenswürdig und korrekt sind. Es stellt sicher, geschützte Daten nicht von unauthorisierten Personen verändert oder gelöscht werden können. Das kann z.B. mit einer digitalen Signatur sichergestellt werden. Es muss nachvollziehbar sein, von wem die Daten geändert wurden, z.B. mit einem Versionierungstool. Durch diese Aufzeichnung ist es möglich, ältere Dateien wiederherzustellen. Es ist außerdem wichtig, die Integrität mittels Permissons zu schützen. Einige Daten sollten eine Checksam für die Verifizierung der Richtigkeit beinhalten.
Verfügbarkeit, garantiert den sicheren Zugang zu Informationen durch authorizierte Personen. Sozusagen müssen gewünschte Funktionen immer zur Verfügung stehen. Dazu ist es notwendig, die Hard- & Software immer auf den neusten Stand zu halten. Wichtig sind auch ein Recovery Plan, für das worst-case scenario zu erstellen. Zusätzliche Tools, wie eine Firewall oder ein Proxy gewährleisten die Verfügbarkeit. Um sich vor Datenverlust zu schützen, müssen die Kopien in verschiedenen, geografischen Standorten gebackupt werden.[9]

Security ist in IoT ein sehr wichtiger Punkt, da alle Geräte mit dem Internet verbunden sind. Es stellt das CIA vor eine große Herausforderung da, große Mengen an Daten geschützt werden müssen, sowie der Anzahl an Geräten die Daten versenden und der verschiedenen Datenformaten. Um eine End-to-End Security sicherzustellen, ist auf jeder Schicht ein eigener Sicherheitslayer relevant. Vom Physikal Layer bishin zum Session Layer. Der WLAN-Zugang sollte mittels WPA2 verschlüsselt werden. Unternehmen sollten WPA2 mit 802.1X für die Schlüsselübergabe verwenden. Subnetze oder Software Firewalls sollen Gefahren minimieren.[10]

D. Storj.io

Die Cloud-Storage Plattform bietet Usern weltweit an ihre Applikationen auf geschützten Speicher zu deployen. Wäre eine gute Variante, die große Menge an Daten, billig und sicher zu speichern. Für die Sicherheit sorgt eine public/private-Key Verschlüsselung und eine kryptographische Hash-Funktion. Es bietet den Vorteil, dass es kostengünstiger und schneller als andere

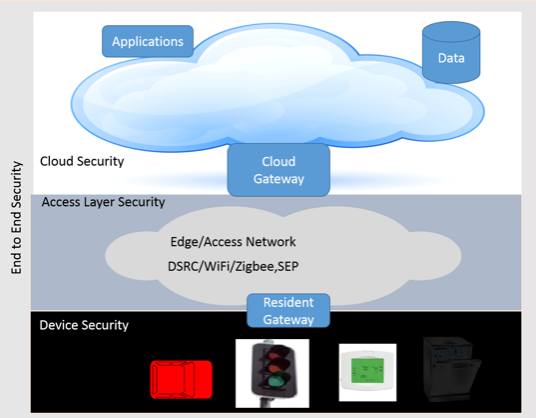

Fig. 5. End-to-End Security [10]

Anbieter ist. Um die beste Sicherheit zu garantieren, sollten die Dateien schon vom Client verschlüsselt werden. Dadurch ist nur der Enduser in der Lage, auf die Dateien zuzugreifen. Jede Datei wird in mehrere Junks aufgeteilt, welche verschlüsselt im Storj Netwerk verteilt werden.
Storj besteht aus zwei Systemen, die kombiniert wurden, um die Cloud-Plattform zu erstellen. MetaDisk ist die erste Applikation, welche dem Anwender erlaubt, Dateien in den Speicher hochzuladen. DriveShare, ermöglicht den nicht benötigten Festplattenspeicher auf die Cloud zu verlagern. Für das Verleihen des Speichers verdient man SJCX, welche für Bitcoin-Transaktionen verwendetet werden kann. Der Verleiher weiß weder welche Dateien auf der Festplatte gespeichert werden, noch kann er darauf zugreifen. Die Plattform gibt dem User dieMöglichkeit das Redundanz Level anzugeben, um zu entscheiden, auf wie vielen Nodes die Applikation gehostet werden soll. Standardmäßig wird auf drei Nodes verteilt.[11]

IV. AUTHENTICATION, AUTHORIZATION, ACCOUNTING

Man muss immer wissen, mit wem man gerade kommuniziert. In der heutigen Zeit ist es umungänglich, sich zu authentifizieren. Für die sichere Übertragung von Daten im Internet sorgt der Secure Sockets Layer(SSL). Wenn Sensoren Daten an die Middleware oder das Gateway die Daten an die Cloud weiterleitet, ist es erforderlich sich zu authentifizieren. Auch wenn man als Benutzer Informationen abfragt, ist eine Registrierung notwendig. Dieses Konzept wird mittels Security-Tokens geregelt. Wenn sich ein Akteur bei einem anderen authentifiziert, wird vor dem Request ein gültiger Token angehängt. Auch bestehende, standardtisierte

Frameworks unterstützen diesen Prozess. Im folgenden Kapitel werden zwei Frameworks, sowie einige Konzepte vorgestellt, um eine sichere Authentifizierung und Übertragung zu gewährleisten. [12]

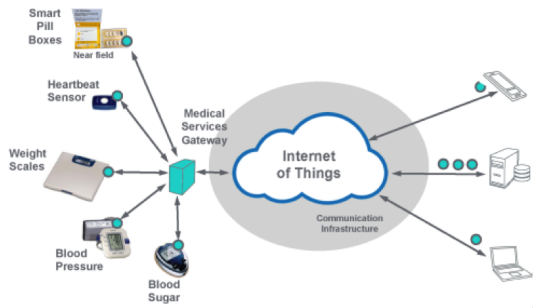

Fig. 6. IoT Communication [12]

A. *Identitätsmanagement OAuth*

Bei einer M2M, Wi-Fi oder Bluetooth Kommunikation von einem Gerät zum anderen, ist ein Vertrauenslevel erforderlich. Ist das Gerät berechtigt, mir Informationen zu senden? Vielleicht sendet ein Hacker die Informationen an den Herd, sich aufzudrehen? OAuth 2.0 ist ein standardisiertes Framework für die Auth- und Authorization.

OAuth ist ein Open-Source Standardframework, welches Zugang zu einer Ressource erlaubt, indem es eine Interaktion zwischen einem Provider und dem Eigentümer herstellt. Bei diesem Prozess bekommt der Anwender, ohne Angabe der Anmeldeinformationen(User ID, Passwort), Zugriffsrechte auf die Ressourcen. Es gibt drei Rollen, den Besitzer, den Server und den Client.
Der **Resource Owner** ist die Person, die der Anwendung das Recht gibt, auf den Account zuzugreifen. Diese Rechte sind beschränkt, also nur auf einen gewissen Bereich und auch nur bestimmte Rechte, lesen oder schreiben.
Die **API** bestehend aus Resource und Authorization Server, gibt dem User mittels einem Token das Recht auf die Dateien, die geschützt hinter dem Resource Server liegen, zuzugreifen.
Application oder der Cliebt möchte den Zugriff auf die Ressource, wofür die Autorisierung von der API bestätigt werden muss.
Bevor man OAuth verwenden kann muss man die Application beim Service registieren. Dies wird durh ein Anmeldeformular in der API durchgeführt, wo man den Namen der Application die Webseite und die *Redirect URI* oder das *Callback URL* angibt.[13], [14]

Im ersten Schritt bekommt der Besitzer der Ressource eine Anfrage, in Form der Zugangsdaten vom Client. Diese Anfrage geht zuerst an einen authorisierten Server und dann an den Besitzer. Für den Zugriff auf den Ressourcenserver benötigt es ein bestimmtes Token, den er entweder vom Authorisierungsserver oder direkt vom Ressourcenserver bekommt.[13]

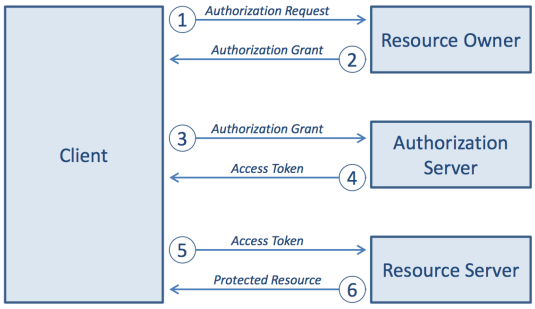

Fig. 7. Kommunikation zwischen Client und Ressource-Server über OAuth [13]

B. *Zwei-Faktor-Authentifizierung*

Es ist wird immer wichtiger den Zugang für das Lesen von sensiblen Daten zu schützen. Nur ich als Hauseigentümer habe das Recht auf mein Zuhause zuzugreifen. Ein normale Anmeldung auf einer Webseite, wo nur der Benutzername und das Passwort verlangt werden, bezeichnet man als Ein-Faktor-Authentifizierung und ist leicht von Hackern zu knacken. Bei einer Zwei-Faktor-Authentifizierung benötigt der User zwei von drei Zugangsdaten um sich anzumelden. Diese drei Attribute können sein:

- Etwas was du weist, z.B. ein Passwort
- Etwas was du immer bei dir hast, z.B. ein Handy oder eine Karte
- Etwas was du bist, wie deine Stimme oder Fingerabdruck

Bei der Anmeldung bei einer Online-Bank gibt man zuerst sein Passwort an, enthält dann einen digitalischen Code auf das Handy, der nur für eine bestimmte Zeit gültig ist und erst dann kann man sich anmelden. Hacker sind vielleicht in der Lage Cookies, Tokens oder eine Session zu übernehmen, aber in ein 2FA Zugang zu bekommen, ist eher schwieriger. 2FA schützt uns nicht vor allen Angriffen, es ist aber sicherer mit, als ohne. So wird gewährleistet, dass niemand anderer als ich auf die Senordaten zugreifen kann. Eine Möglichkeit den Account zu hacken wäre, eine Konto-Wiederherstellung,

welche das Passwort zurücksetzt und man sich nochmal anmelden muss. Es wird vorgetäuscht, dass jemand Zugriff auf deine Daten hat, obwohl er 2FA benutzt. Das zurücksetzen des Kontos, deaktiviert 2FA, wodurch man nun in der Lage ist sich normal anzumelden. Eine Möglichkeit diese Problem zu beseitigen wäre die Verwendung von biometrischen Daten. Nachteilig könnten sein, bei Ausfall des Handyakkus, keine Nachrichten vom Provider empfangen werden können. Geht das Handy verloren, kann dies ebenfalls zu Problemen führen.[13]

C. Zertifikate (Zertifizierungsstelle)

Die Nutzung von digitalen Zertifikaten ermöglicht die sichere, vertrauenswürdige Interaktion zwischen den Geräten über das Netzwerk. Die Public-Key-Infrastruktur erstellt, verteilt und überprüft digitale Zertifikate. IoT sind mehr gefährdet als herkömmliche Geräte, wie PCs oder Laptops, weil die Firmware ohne Sicherheit entwickelt wurde. Es stellt sich als Schwerigkeit dar, sich zu authentifizieren, wenn die Geräte über M2M kommunizieren.
Bei einer digitalen Zertifizierung bekommt jedes Gerät einen Private Key und einen Public Key, um andere Geräte zu erreichen. Das kann sich jedoch als schwierig erweisen, wenn das Netzwerk viele IoT beinhaltet. Dieses Problem kann durch einen Hub beseitigt werden, der die Private Keys in den Subnetzen verwaltet. PKI sorgt auch dafür, dass die Keys erneuert oder verworfen werden können.[15]

1) Kostenpflichtige Lösungen: Mögliche Zertifizierungsstellen sind Symantec, Thawtem GeoTrust und RapidSSL.

2) Gratislösung: Let's Encrypt bietet kostenlose Zertifikate die in allen gängigen Webbrowsern als vertrauenswürdig eingestuft werden. Dafür ist es notwendig, einen Encrypt-Client von dem Anbieter auf seinem Server zustarten. Im Hintergrund kontaktiert der Server die CA und weist sich via Domain Validaiton aus, um das Zertifikat zu bekommen.[16]

V. Desaster Recovery

Disaster Recovery (dt. auch Katastrophenwiederherstellung), im Folgenden auch DR genannt, beschreibt die Vorbereitung und Reaktion auf sogenannte Katastrophen, die abgespeicherte Daten und Lauffähigkeit eines IT-Systems betreffen. In diesem Bereich der Sicherheitsplanung ist mit negativen Ereignissen all das gemeint, was den Betrieb eines Unternehmens gefährdet.

A. Disaster Recovery Plan

Ein DRP dokumentiert konkrete Richtlinien, Verfahren und Maßnahmen, um die Störung eines Unternehmens im Falle eines Desasters zu begrenzen, und möglichst innerhalb eines bestimmten Zeitrahmens wieder zurück in den Normalzustand zu gehen.

1) Szenarien: Mögliche Katastrophen könnten sein:

- Softwareausfälle
- Stromausfälle, Brand oder Wasserschaden
- Sicherheitsprobleme

2) Vorbeugemaßnahmen: Softwareausfälle können immer wieder vorkommen. Um sich dagegen vorzubeugen, empfehle ich eine Active-Active Strategie. Das bedeutet wenn ein System ausfällt, kann das wartende Ersatzsystem den Part übernehmen.
Um sich am Besten gegen Stromausfälle, Brand oder einen Wasserschaden zu schützen, schlage ich vor alle Daten auszulagen, sei es in die Cloud oder Rechenzentrum. Dort sind die Daten vor Naturkatastropen geschützt. Durch regelmäßige Backups, Replikationen und Hot Swap sind die Daten sicher und immer verfügbar, sodass unser System keine Downzeit hat.
Sicherheitsprobleme können nur dann stattfinden, wenn unsere Umgebung zu wenig geschützt ist. Es ist immer wichtig, die Daten verschlüsselt zu übertragen und mit Zertifikaten zu sichern. Um Unauthorisierten den Zugang zu verwähren, sind Authentisierungssoftwaren unumgänglich. Dank der Zwei-Faktor-Authentifizierung wird Hackern erschwert, sich einen Zugang zu verschaffen.

3) Reaktion: Sollten die Vorbeugemaßnahmen nicht ihren Zweck erfüllen, was sehr unwahrscheinlicht ist, müssen wir uns ernsthaft Gedanken machen.

B. Hot Swap/Hot Standby

Unter How-Swap versteht man, den Austausch von Systemkomponenten während den laufenden Betrieb. Dabei soll es zu keiner Betriebsunterbrechung, sowie zu keinen Neustart oder Neukonfiguration kommen. Die Hardware muss so aufgebaut sein, dass bei einem Austausch keine Konfiguration notwendig ist. Die Konfiguration muss zuvor von der alten Hardware abgespeichert werden und von der neuen dann automatisch übernommen bzw. geladen werden.[17]
Bei Hot-Standby wird eine redundante Komponente in den Wartzustand versetzt. Sie wird erst dann aktiv, wenn die primäre Komponente ausfällt.

Dieser Verbund an fehlertoleranten Geräten, auch Verfügbarkeitsverbund genannt, erhöht die Verfügbarkeit und die Betriebssicherheit. So kann ich als Anwender sicher sein, dass die Informationen über mein Haus, immer zur Verfügung stehen. Dieser Prozess wird nicht nur durch Hot Standby bezweckt, sondern auch durch das *swappen* auf ein Rechenzentrum. Rechenzentren sind komplex und teuer, doch im Verhältnis zum wirtschaftlichen Schaden, angemessen? Bei Systemen, die leicht wieder hergestellt werden können, reicht auch ein einfaches Backup. Aber was ist, wenn die ganze Infrastruktur lahmgelegt wird?
Was wäre dann die geeignete Notfall-RZ-Architektur?[18]

- **Active RZ**: Ist geeignet um eine maximale Ausfallsicherheit zu gewährleisten. Dahinter steckt meistens ein riesieger Cluster.
- **Hot Standby RZ**: Synchrone Datenspiegelung unter den Rechenzentren. Kann als Hochverfügbarkeits-RZ, sowie gleichzeitig als Notfall-RZ verwendet werden.
- **Warm Standby RZ**: Asynchrone Datenreplikation. Die Serverumgebung sollte dabei voll virtualisiert, die Tests zyklisch und automatisiert sein, um sich Konfigurationsanpassungen beim Umschalten zu ersparen. Ohne Automatisierung dauert die Wiederherstellung ansonsten mehrere Tage. Es kann bei diesem Standby ein maximaler Datenverlust von einem Tag passieren, da die Backups immer in den Nächten stattfinden.
- **Cold Standby**: Hier steht im Notfall nur eine Auswahlmöglichkeit in ein RZ zur Verfügung.

1) Virtualisierung: Ein passiv laufendes System im Hintergrund mitlaufen zu lassen, ist ziemlich teuer. Mit der Virtualisierung kann die Ausfallsicherheit erhöht und der Preis minimiert werden. Von VmWare wird das Cluster-Konzept in die virtuelle Welt übertragen. Bei einem Ausfall wird die VM einfach auf einem anderen Server, ausgestattet mit einem Hypervisor, gestartet. Bei einem normalen Cluster können Änderungen Tage dauern bis diese auf der Gegenseite übernommen werden.
Dank der Virtualisierung braucht die DR-Seite keine Hardware mehr. Es braucht keine 1-zu-1 Beziehung der Serverhardware , also muss man keine zwei gleichen Server kaufen. Auf der aktiven Stelle werden mehrere virtuelle Hosts erstellt, die die Verfügbarkeit erhöhen.
VMware Site Recovery Manager(SRM) wurde designt um virtuelle Umgebung zwischen den ESX Clustern zu maßstabsgetreu und standortübergreifend zu verschieben und wiederherzustellen. Es ist für Unternehmen gedacht, die ihre eigene Private Cloud für die DR ausbauen möchten. Workflows und regelmäßige Tests sichern die Verlässlichkeit der DR. [19]

C. Cluster Heartbeat

Bei einem Failover-Cluster arbeiten zwei Server synchron. Der eine ist aktiv, während der andere im Standby-Modus ist und den anderen beobachtet. Er wartet darauf, dass der erste Server ausfällt, damit er einspringen kann. Diese ständige Überwachung oder Abfrage funktioniert mittels einem Heartbeat. Das kann z.B. bei beim Hot-Standby, wenn ich meine Daten auf ein anderes Rechenzentrum verlagere, der Fall sein. Aber auch bei der Virtuallisierung muss getestet werden, dass alle Hosts noch lauffähig sind. Dieses Kontrollsignal testet die Funktionalität des Masters. Die Übertragung findet meist über Ethernet, Fibre Channel oder Nullmodem-Kabel statt.
Bei Unterbrechung der Heartbeat-Übertragung, schalten sich beide Server als Primärknoten ein, wo sie beide auf den selben Speicher zugreifen. [20]

VI. ALGORITHMEN UND PROTOKOLLE

In diesem Kapitel sprechen wir Algorithmen an, die für eine sichere Kommunikation zwischen zwei Endgeräten ausgelegt sind. Dabei wurde darauf geachtet, dass die Techniken stromsparend agieren, da die Geräte wenig Power haben. Auch die Hochverfügbarkeit spielt eine wichtige Rolle, die mittels Load Balancing Algorithmen abgedeckt wird. Neben der Sicherheit, kann es häufig vorkommen, dass Systeme ausfallen. Um dem entgegenzuwirken, ist eine Datenreplikation notwendig, die mittels XDCR realisiert wird.

A. Lastenaufteilung

Der Content Switch, wo ein Server nur für einen Content zuständig ist, benötigt Protokoll um Anfragen vom Client oder neue Daten von Sensoren entgegenzunehmen und gleichmäßig auf den Datenbank-Cluster zu verteilen. Dafür eignen sich im Speziellen drei Algorithmen. Load Balancer können verschiedene Algorithmen verwenden.

- **Round-Robin**: Ein LB weist jedem Server der Reihe nach eine Verbindung zu. Eignet sich gut, wenn aus vielen Servern gewählt werden kann, da der Algorithmus wenig Rechenzeit benötigt.

- **Least Connections**: Jede neue Anfrage wird an den Server, mit den geringsten, gleichzeitig aktiv vorhandenen Verbindungen zugesandt.
- **Weighted Distribution**: Jeder Server bekommt aufgrund seiner Hardware eine Gewichtung von 5-1 zugeteilt. Je höher die Gewichtung, desto mehr Anfragen gehen an den Server.

B. Web Services

Ein Webservice ist ein Verfahren für die Kommunikation zwischen Geräten. Es gibt zwei Arten von Webservices: SOAP und REST.

1) SOAP: Das standardisierte Simple Object Access Protocol definiert ein Kommunikationsprotokoll basierend auf XML. Es verwendet die Transportprotokolle HTTP und SMTP. Mithilfe von SOAP können Daten zwischen Systemen ausgetauscht werden. Es regelt, wie Daten in der Nachricht abzubilden und zu interpretieren sind. Zum Senden von Daten können die Transportprotokolle FTP, SMTP oder HTTP/S verwendet werden. Die End-to-End Verschlüsselung kann nur mit dem WS-Security Standard erfolgen. [21]

2) REST: Representational State Transfer beschreibt eine Menge an Architekturprinzipien, basierend auf dem HTTP-Protokoll. Es kann dazu verwendet werden Daten von Geräten abzufragen und darzustellen. REST ist auch schon in CoAP, Kapitel I-B2, implementiert. Mittels CoAP kann der Client an das Gerät Befehle übergeben und auch Daten abfragen. Über HTTP ist es einem Client möglich, entkoppelt Nachrichten zu senden und zu empfangen. Auf die Daten kann mittels einer URI zugegriffen werden. Über REST können verschiedene Systeme kommunizieren. Der RESTful Web Service ist stateless. Das bedeutet, dass sich der Server nur fur die Dauer eines Requests fur den Client interessiert und danach jegliche Informationen verwirft, die er über diesen Client hat. Dadurch gibt es keine Bindung zwischen Server und Client. Jegliche Informationen am Client müssen aufbewahrt werden. Außerdem ist dieses System skalierbarer. Requests können von verschiedenen Serverinstanzen bearbeitet werden.
Die REST-Schnittstelle unterstützt Daten-Formate wie, JSON, XML, CSV oder HTML. Eines der bekanntesten Frameworks ist Spring. [21]

C. Verschlüsselungsalgorithmus DTLS & Ipsec & LWC

Das folgende Kapitel befasst sich mit Protokollen und Algorithmen für die sichere Übertragung. Dabei wird darauf geachtet, dass die kleinen Geräte die Anwendungen hinsichtlich des Energieaufwands umsetzen können.

1) DTLS: IoT bevorzugen das verbindungslose Netzwerkprotokoll UDP, um die Paketgröße und den Energieverbrauch zu reduzieren. Eine TCP-Verbindung in einem WLAN funktioniert nicht so gut mit Geräten, die eine geringe Stromversorgung, sowie eine verlustbehaftete Verbindung aufweisen. Eine Kombination zwischen UDP und TLS stellt die adaptierte Version DTLS dar. Es löst zwei Probleme, dass Ordnen der Pakete und das Verlieren von Paketen. DTLS vermeidet Cryptografischen Overhead. Um geheimzuhaltenden Daten sicher zu übertragen, verwendet CoAP das DTLS, als Sicherheitsprotokoll, für die Authentifikation und Kommunikation zwischen den Geräten. Es kümmert sich um den Schlüsselaustausch und die Verschlüsselung der Daten. DTLS benötigt einige Nachrichten um eine sichere Session zu erzeugen, den in der Entwicklungszeit war es für Nachrichten gedacht, deren Länge unbenkannt ist. Das Protokoll ist in der Lage Session-Keys zu erneuern und diese in einem 6LoWPAN zu managen. Während der Nutzzugangsphase verteilt der 6LBR-6LoWPAN-Router einen L2 Schlüssel für das Authentifizierungsverfahren.
In unserem Fall komprimiert das 6LoWPAN den DTLS-Header, was zu einer Energieeffizientz führt und die 6LoWPAN Fragementierung verhindert, welche gern als Angriffsziel genommen wird.
DTLS kann dazu verwendet werden, neue Geräte zu dem Netzwerk hinzuzufügen. Dies passiert entweder mit dem *Raw Public Key* oder *Public Key Zertifikat*, welches einen neuen sicheren Kanal mit dem hinzugefügten Geräte erzeugt. Der Aufbau des DTLS ähnelt sehr dem TLS. Es besteht aus zwei Schichten. Die Erste beinhaltet das Record-Protokoll und obere Schicht des Handshake-, Alert- und ChangeCipherSpec-Protokoll. Das Record-Protokoll ist für die Vertraulichkeit, Integrität und Authentizität, die Reihenfolge der Sequenznummern, sowie für die Verschlüsselungsmechanismen zuständig. Außerdem kümmert es sich um drüberliegende Protokoll. Das Handshake Protokoll, kümmert sich, dass beim Empfänger die Paktete in der richtigen Reihenfolge sind und falls es zu einem Fehler kommt, das Paket neu beantragt wird.[5]
DTLS kann nicht alle Verschlüsselungsalgorihtmen verwenden, z.B. RC4 ist ungeeignet. Das Blockverfahren CBC Mode kann verwendet werden um die Nachricht zu verschlüsseln. Der *Initialization Vec-*

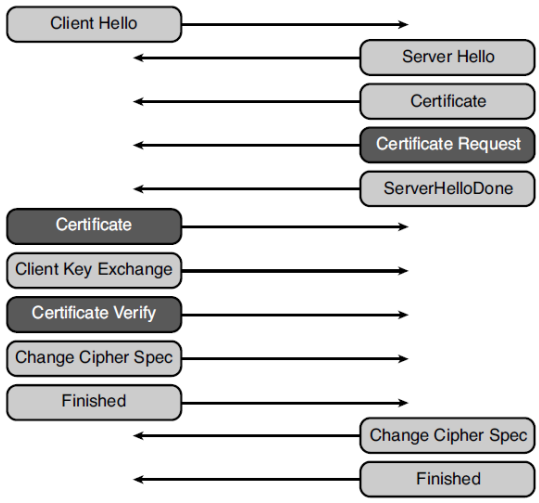

Fig. 8. DTLS Handshake Prozess [5]

tor(IV) stellt unseren Geheimschlüssel dar, mit dem wir den Klartext verschlüsseln. Es gibt eine definierte Blockgröße, in der wir den Text aufspalten. Jeder Block wird mittels dem AES oder DES Verfahren verschlüssel, sodass ein Blockchiffre entsteht. Ab der zweiten Runde wird der erzeugte Ciphertext mit dem Klartextblock XOR-verknüpft, AES und haben wieder einen Geheimtext.[22]

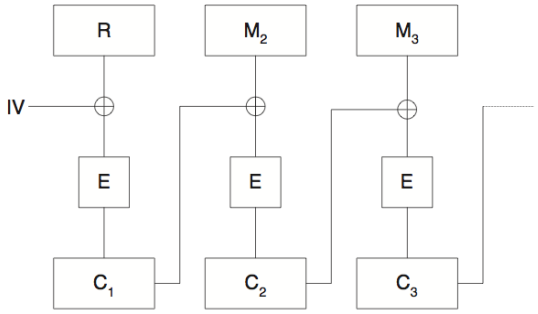

Fig. 9. DTLS Handshake Prozess [22]

2) IPsec: Das Design von DTLS und IPsec sind sehr nahe, doch DTLS unterscheidet sich in zwei Aspekten. DTLS ist ein Application Protokoll, dadurch ist es einfacher in Anwendungen zu integrieren. Weiters verwendet es das TLS-Progrmmierungsmodel, indem Sicherheitskontexte Anwendungen kontrollieren. IPsec hat keine standardisierte API odein Programmiermodel, die in der Implementierung verbreitet sind. Im Vergleich dazu ist das IPsec Schlüsselmanagement so komplexer als TLS.
IPsec kann für die Kommunikation zwischen den Knoten in einem 6LoWPAN, Kapitel I-B1, Netzwerk verwendet werden. Es dient zum Schutz der die host-to-host, network-to-network und network-to-host Kommunikation. IPsec unterstützt die Vertraulichkeit, Integrität, Authentifikation, Kapitel III-C, und Schutz vor Angriffen, für jedes IP-Paket. Solche Sicherheitsdienste werden von zwei IPSec-Sicherheitsprotokolle, Authentication Header(AH) und Encapsulated Security Payload(ESP) implementiert. AH kümmert sich um die Integrität, Authentifikation, während ESP zusätzlich auch für die Vertraulichkeit zuständig ist. "*IPSec AH and ESP define only the way payload data (in clear or enciphered) and IPSec control information are encapsulated, while the effective algorithms for data origin authentication/integrity/confidentiality can be specified separately and selected amongst a set of available cipher suites.*"[23] Um die Verschlüsselung kümmert sich das IPSec Security Association (SA), welches eine Sammlung von Verschlüsselungsalgorithmen anbietet. Andernfalls kann auch der IPSec Internet Key Exchange (IKE) Protokoll zum Einsatz kommen, welches die asymetrische Verschlüsselung benutzt. So komplexe Verschlüsselungsalgorithmen sind aufgrund der begrenzten Ressourcen ungeeignet. Geeigneter ist LWC, der weniger Engerie und Power verbraucht.[23]

3) Blockchiffren: Blockchiffren teilen die Nachricht, welche verschlüsselt werden soll, in eine fixe Anzahl an Blöcken.

a) Tiny Encryption Algorithm (TEA): TEA operiert mit zwei 32-bit unsigned Integers, eine Blockgröße von 64-Bit und einem 128-bit großen Schlüssel. Die Nachricht wird in 64-Bit Blöcke umgewandelt, die addiert, XOR-verknüpft und geshiftet werden.
Da einige Schwächen auftraten, kam es zu Weiterentwicklungen, unten den Namen XTEA oder XXTEA. Durch die leichten Operationen und den kleinen Codegröße eignet sich die TEA-Familie perfekt für das IoT. [23]

4) Public-Key: Das Public-Key verfahren ist für die sichere Kommunikation ausgelegt. Da RSA für die Verschlüsselung einen hohen Leistungsaufwand, sowie einen großen Schlüssel benötigt, wäre die Alternative ECC vom Vorteil.[23]

a) Elliptic Curve Cryptography (ECC): ECC bietet im Gegensatz zu RSA eine höhere Sicherheit und bessere Leistung. Es basiert auf der algebraischen Struktur elliptischer Kurven über endlichen Körpern. Aufgrunddessen, dass die gleiche Sicherheit wie beim RSA, nur mit kürzerer Schlüssellänge erzielt werden

kann und leichtere Operationen eingesetzt werden, eignet es sich für embedded Devices.[23]

5) Vergleich: Die symetrische Verschlüsselung wird aufgrund der Verarbeitungsgeschwindigkeit, dem Rechenaufwand und der Größe der übertragenen Nachrichten bevorzugt. Das Public-Key-Verfahren eignet sich dazu, den symetrischen Schlüssel zum anderen Endpoint zu übertragen, um eine verschlüsselte Kommunikation einzuleiten.[23]

D. Cross Datacenter Replication (XDCR)

XDCR ist ein asynchrones Datenreplikationssystem. Es bietet eventuelle Konsistenz von Daten zwischen zwei Couchbase Clustern an. Cross Datencenter Replikation betrifft die Replikation von aktiven Daten zu geografisch entfernten Datencentern, für das DR. Für das CAP-Theorem, Kapitel VII-C, entspricht XDCR dem AP, da die Daten nur eventuell konsistent und hochverfügbar sind. Hätte man nur einen Cluster würde die Verfügbarkeit darunter leiden.[24]

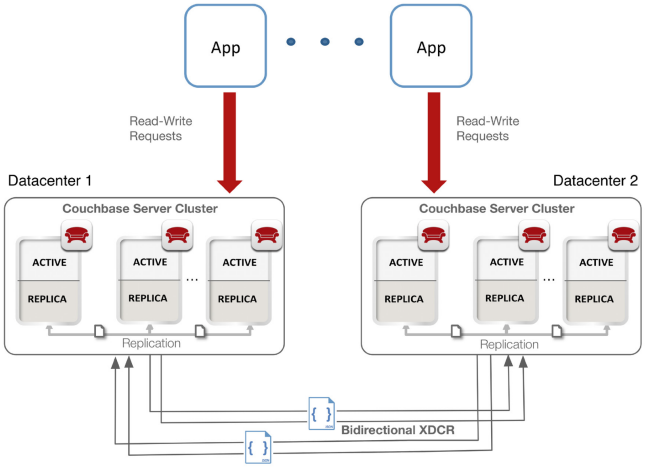

Fig. 10. Cross Replikation [24]

XDCR unterstützt zwei Arten.
Die einseitige(*unidirectional*) Replikation. Hier werden die aktiven Daten, auf einen Backup-Cluster repliziert. Die zweiseitige(*bidirectional*) Replikation. Eignet sich für Load Balancing, da auf beiden Seiten, die gleichen Daten sind. Das hat den Vorteil, dass die Daten näher beim Kunden und dadurch schneller zugreifbar sind.[24]

VII. KONSISTENZ UND DATENHALTUNG

Der Begriff Big Data gewinnt immer mehr an Bedeutung. Die Menge, sowie die Vielfältig der Daten wird immer größer. Das Managen von solchen Daten, die von verschiedenen Geräten kommen, erfordert ein neues Level an Flexibilität, Agilität und Skalierbarkeit. Welche Systeme sind am besten für die dynamische Datenspeicherung geeignet?

A. NoSQL

Für die Datenhaltung von IoT beweisen NoSQL-Datenbanken ihren Stellenwert. RDBMS sind nicht immer die beste Lösung für alle Situationen, da es nicht mit unstrukturierten, unerwarteten und der Menge an Daten zurecht kommt. Im Thema Geschwindigkeit und Skalierung ist NoSQL auch vorne. Um SQL-Datenbanken zu skalieren müssen die Datenbankpartitionen aufwändig repliziert werden.
NoSQL steht für Not Only SQL und managt Daten, die nicht umbedingt die Struktur einer relationalen Datenbank hat. Die Daten können als Objekt z.B. als Graph, Key-Value oder als Dokument dargestellt werden. Meistens macht es keinen Sinn, Sensordaten in eine Tabelle zu speichern. Laut dem CAP-Theorem Kapitel VII-C, kann NoSQL nur zwei Merkmale unterstützen. Da NoSQL nicht so sehr auf die Konsistenz acht nimmt, entspricht das ein AP. Zusammenfassend bietet NoSQL eine erhebliche Flexibilität in Bezug auf die Datenverwaltung. [25]
Die NoSQL-Plattform Cassandra passt in dieses Schema gut hinein. Die Datenbank, basierend auf Key-Value ist so designt, viele Daten aufeinmal aufzunehmen. Cassandra besteht nicht aus einem Server, sondern aus einem Cluster, wo sich jeder Knoten um eine Transaktion kümmert. Fällt ein Knoten aus, sind die anderen noch immer in der Lage, Anfragen aufzunehmen. Also entsteht kein Datenverlust.
Andere Mögliche System wären HBase und Couchbase.

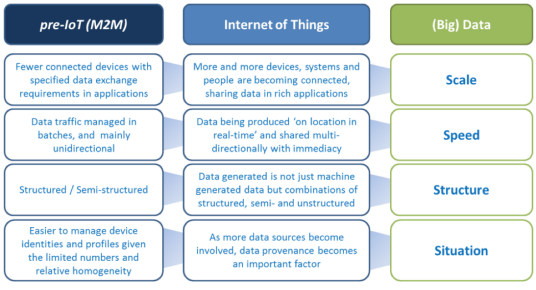

Fig. 11. Anforderungen an die Systeme [25]

B. Zukunftaussichten

In Zukunft werden die Datenspeicher nicht physisch, sondern biologisch sein. DNA-Festplatten, die ohne

Strom betrieben werden, ermöglichen enorme Mengen an Daten auf kleinsten Raum zu speichern. Außerdem bieten sie den Vorteil, dass die Daten über mehrere Tausend Jahre beständig sind. Bei der Speicherung werden die Daten, in kleine Teilstücke zerlegt und nummeriert. Anhand der Nummerierung können die Daten wieder hergestellt werden. Da die DNA so klein ist, können Daten leicht transportiert werden, jedoch ist Preis derzeit noch zu hoch, um es in der Praxis durchzusetzen.[26]
Außerdem haben britische Wissenschaftler ein neues langlebiges Speichermedium entwickelt, die 5D-Glasscheibe. Sie ist so groß wie eine zwei Euro Münze und bietet eine Speicherplatz von 360 Terabyte. Laut Angaben soll die Platte knapp 13.8 Millarden Jahre halten. "*Die hohe Speicherdichte wird über Nanostrukturen erreicht, die in Quarz eingeprägt werden. Die Daten werden mit einem schnellen Femtolaser auf die Glasplatte übertragen. Dadurch entstehen winzige Punkte im Quarz, die nur fünf Mikrometer auseinander liegen. Licht, das diese Strukturen durchquert, wird dabei polarisiert. Das ermöglicht das Auslesen mit einem optischen Mikroskop und einem Polarisationsfilter.*"[27]

C. CAP-Theorem

Das CAP-Theorem beschreibt das Verhältnis zwischen Consistency, Availability und Partition Tolerance (Konsistenz, Verfügbarkeit und Partitionstoleranz) indem es besagt, dass davon nur zwei in einem verteilten System gleichzeitig angeboten werden konnen.[28]

1) Consistency: Zählt zu den Sicherheitseigenschaften und beschreibt im Groben, dass eine Anfrage eine richtige Antwort erhält. Um dies zu Erhalten, benötigt man eine Reihenfolge fur alle Operationen, sodass zu einem Zeitpunkt nur eine Operation fertiggestellt wird. Somit lesen alle Knoten zur selben Zeit die gleichen Daten.[28]

2) Availability: Die hohe Verfügbarkeit ist in einem Smart Home ausschlaggebend. Es muss auf jede Anfrage eine Antwort zurückkommen.[28]

3) Partition Tolerance: Trotz Datenverlust oder Systemausfall, funktioniert System noch weiter. Partioniert man das Netzwerk, verliert man entweder an Consistency, weil man Änderungen auf beiden Partitionen erlaubt oder Availability, weil man einen Fehler entdeckt und man das System abstellen muss um dies zu richten.[28]

VIII. CONCLUSION

Ein Smart Home birgt viele Vorteile, aber um sicherzustellen, dass alles reibungslos läuft, muss auf viele Details im Hintergrund geachtet werden. Die Kommunikation zwischen den Geräten über CoAP, die sichere Übertragung mit DTLS, die Authentifizierung mit OAuth oder 2FA. Auch Backups und Replikationen sind notwendig, um die Hochverfügbarkeit und Ausfallsicherheit zu gewährleisten. Man sieht, dass eine geeignete Infrastruktur, das Zusammenspiel von passenden Systemen, Protokollen und Strategien, der Schlüssel für ein sicheres und gut funktionierendes Smart Home ist.

LIST OF FIGURES

REFERENCES

[1] B. C. Pühringer, "Cloud computing for home automation." https://www.auto.tuwien.ac.at/bib/pdf_TR/TR0167.pdf, 2016.
[2] Kemp, "Layer 7 load balancin." https://kemptechnologies.com/load-balancing/layer-7-load-balancing/, 2016.
[3] L. Deru, "6lbr." https://github.com/cetic/6lbr/wiki, 2016.
[4] E. Lehmann, "Grundlagen 6lowpan." https://www.dresden-elektronik.de/fileadmin/Downloads/Dokumente/white_paper/fundamentals_6lowpan-WP-de.pdf, 2012.
[5] N. Modadugu, "Secure coap using enhanced dtls for internet of things." https://crypto.stanford.edu/ nagendra/papers/dtls.ps, 2010.
[6] A. Piper, "Mqtt, das m2m und iot protokoll." https://www.predic8.de/mqtt.htm, 2016.
[7] E. Rahm, "Datenbankverteilung." http://dbs.uni-leipzig.de/buecher/mrdbs/mrdbs-34.html, 1994.
[8] P. Mah, "How to build a storage and backup strategy for your small business." http://www.cio.com/article/2378019/small-business/how-to-build-a-storage-and-backup-strategy-for-your-small-business.html, 2014.

[9] TechTarget, "confidentiality, integrity, and availability (cia triad)." http://whatis.techtarget.com/definition/Confidentiality-integrity-and-availability-CIA, 2016.

[10] R. Syamala, "Is security an afterthought in internet of things." http://lochbridge.com/blog/is-security-an-afterthought-in-internet-of-things/, 2015.

[11] Storj, "Frequently asked quesstions." https://storj.io/faq.html, 2016.

[12] AVISIAN, "Authentication in the iot - challenges and opportunities." http://www.secureidnews.com/news-item/authentication-in-the-iot-challenges-and-opportunities/, 2016.

[13] T. Borgohain, "Authentication systems in internet of things." http://www.ijana.in/papers/V6I4-11.pdf, 2015.

[14] M. Anicas, "An introduction to oauth 2." https://www.digitalocean.com/community/tutorials/an-introduction-to-oauth-2, 2014.

[15] B. Tarzey, "Securing the internet of things - time for another look at public key infrastructure (pki)?." http://www.computerweekly.com/blog/Quocirca-Insights/Securing-the-Internet-of-Things-time-for-another-look-at-Public-Key-Infrastructure-PKI, 2015.

[16] D. Schirrmacher, "Let's encrypt startet öffentliche beta: Kostenlose ssl/tls-zertifikate für jedermann." http://www.heise.de/security/meldung/Let-s-Encrypt-startet-oeffentliche-Beta-Kostenlose-SSL-TLS-Zertifikate-fuer-jedermann-3031699.html, 2015.

[17] TechTarget, "hot swap." http://whatis.techtarget.com/definition/hot-swap, 2016.

[18] W. Miedl, "Welche notfall-architektur wo sinnvoll ist." http://www.computerwoche.de/a/welche-notfall-architektur-wo-sinnvoll-ist,3069974, 2014.

[19] G. Crump, "Improving disaster recovery with vmware site recovery manager and infrastructure virtualization." http://searchvmware.techtarget.com/tip/Improving-disaster-recovery-with-VMware-Site-Recovery-Manager-and-infrastructure-virtualization, 2008.

[20] ITWissen, "Failover-cluster." http://www.itwissen.info/definition/lexikon/Failover-System-fail-over-system.html, 2014.

[21] S. Dhingra, "Rest vs. soap: How to choose the best web service." http://searchsoa.techtarget.com/tip/REST-vs-SOAP-How-to-choose-the-best-Web-service, 2013.

[22] A. A.Chavan, "The design and implementation of datagram tls." http://www.rroij.com/open-access/secure-coap-using-enhanced-dtls-forinternet-of-things.pdf, 2014.

[23] S. Cirani, "Enforcing security mechanisms in the ip-based internet of things: An algorithmic overview." www.mdpi.com/1999-4893/6/2/197/pdf, 2013.

[24] Couchbase, "Extend your data tier with xdcr." http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase_WP_Cross_Datacenter_Replication_in_Couchbase_Server.pdf, 2013.

[25] E. Berthelsen, "Why nosql databases are needed for the internet of things." http://s3.amazonaws.com/info-mongodb-com/2014-04-10_machina_research_databases_and_the_iot.pdf, 2014.

[26] Birgit, "Britische wissenschaftler entwickeln dna festplatte" – daten werden in dna molekülen gespeichert." http://www.trendsderzukunft.de/britische-wissenschaftler-entwickeln-dna-festplatte-daten-werden-in-dna-molekulen-gespeichert/2013/01/26/, 2013.

[27] DiePresse, "Mit 5d-glasscheibe daten für milliarden jahre speichern." http://diepresse.com/home/techscience/hightech/4928727/Mit-5DGlasscheibe-Daten-fur-Milliarden-Jahre-speichern-, 2016.

[28] N. A. L. S. Gilber, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services." ACM SIGACT News, Volume 33 Issue 2:51-59, 2002.