Sicherheit im Bereich Smart Home

Seminararbeit IT-Sicherheit

Wintersemester 2017

Eingereicht von:

Sebastian Jung

Winf101597

Betreuer:

Prof. Dr. Gerd Beuster

Datum: 28.05.2018

Inhalt

Einleitung

In dieser Arbeit wird der Samsung Smart Things Hub auf Schwachstellen und Möglichkeiten, diese Schwachstellen auszunutzen, untersucht. Dabei werden vor allem die Sicherheitsarchitektur und der interne Aufbau des Hubs untersucht. Auf Möglichkeiten, Sicherheitslücken im ZigBee oder ZWave Protokoll auszunutzen, wird hier verzichtet. Es wird sich auf die Möglichkeiten die Sicherheitslücken des SmartThings Hub auszunutzen beschränkt und geprüft, inwiefern der Endverbraucher sich vor diesen schützen kann. Abschließend werden die Risiken und Schwachstellen des Smart Hubs mit dem Risikobewusstsein der Endnutzer verglichen.

Als Grundlage der Arbeit wird ein Paper genutzt, welches mit dem Thema „Security Analysis of Emerging Smart Home Applications“ genau den ersten Teil dieser Arbeit abdeckt [1].

Die Anzahl von Smart Home Geräten nimmt immer weiter zu. Sehr viele Hersteller bieten mittlerweile einen eigenen Smart Hub an. Von Amazon Alexa [5] über Google Home [6] oder Samsung Smart Things [7] gibt es viele digitale Assistenten, die das Leben erleichtern sollen. Ebenso nimmt die Zahl der Endgeräte rasant zu. Drahtlos steuerbares Licht, digitale Türschlösser, intelligente Kühlschränke oder Waschmaschinen um nur einige Beispiele zu nennen. Meist bieten smart home Geräte nicht nur einen erhöhten Komfort, sondern locken auch mit Energiesparpotenzial. Das Licht, was sich selbstständig an und aus schalten kann, um so ein bewohntes Zuhause zu suggerieren, verbessert das Sicherheitsgefühl im Urlaub. Ansonsten ließe sich noch eine Kamera einrichten, um bei Bewegungen informiert zu werden und man wäre in der Lage zu prüfen, was in den eigenen vier Wänden gerade passiert.

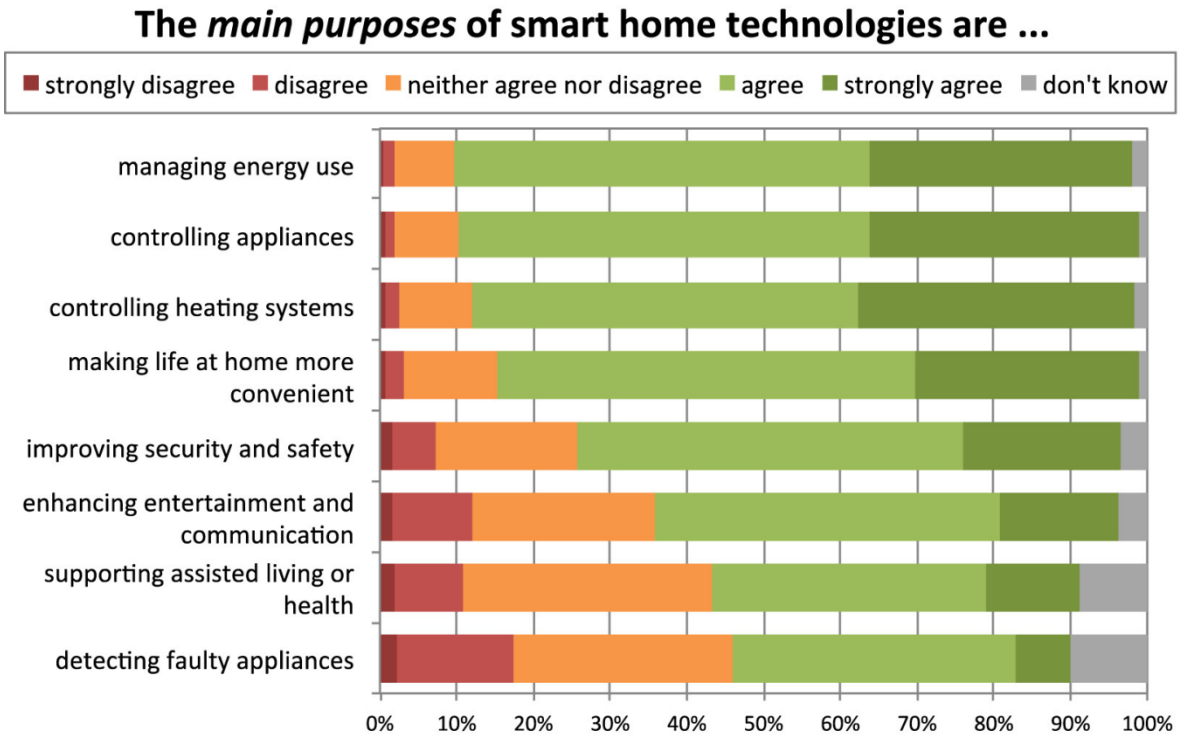

Abbildung 1: Vorteile von Smart Home Technologien[2]

Die Grafik basiert auf einer Online-Umfrage, die in Großbritannien durchgeführt wurde. Hierbei wurden insgesamt 1150 Personen befragt, die älter als 18 Jahre waren.

Im ersten Teil der Umfrage gab es neben Soziodemographischen Fragen auch eine Auslesefrage. Die Absicht hierbei war, dass vorurteilbehaftete Antworten, soweit wie möglich herausgefiltert werden können, da Teilnehmer mit keinem Vorwissen zu SmartHome Technologien nicht mehr an der weiteren Umfrage teilnehmen durften. So wurden von anfänglich 1150 Teilnehmern 125 Teilnehmer herausgefiltert.

Die Hauptvorteile, die von den Teilnehmern gesehen werden, sind das Energiemanagement und die Möglichkeit Geräte zu kontrollieren. Diese beiden Punkte haben eine starke Übereinstimmung. Als ebenfalls wichtiger Grund für ein Smart Home wird das Kontrollieren der Heizungsanlage gewertet. Erst danach kommt die Erhöhung der Sicherheit für das eigene Heim. Erhöhung des Komforts oder die Erhöhung der Sicherheit sind die am häufigsten benutzten Werbetitel für Smart Devices.

Aber eine Verbindung ins Internet bedeutet auch, dass diese Geräte mögliche Ziele von Attacken sein können. Türschlösser, die von einer fremden Person ferngesteuert werden können. Kameras, die benutzt werden können, um einen selbst auszuspionieren. Heizungsaggregate, die auf einmal nicht mehr heizen, oder zu stark heizen, sind jedoch eher Vorstellungen, die von smart home Geräten abschrecken.

Machen diese Geräte das Leben einfacher und sicherer, oder öffnen sie fremden Leuten sprichwörtlich die Tür zum Haus?

Die Grundlage der weiteren Analyse stellt der Samsung Smart Things Hub dar. Hierbei werden vor allem die Schwächen des Frameworks und den daraus resultierenden Effekten für die Apps betrachtet. Hierbei wurde eine Vielzahl von statischen Analysetools verwendet, da Smart Things an sich eine geschlossene Plattform darstellt.

Probleme von Samsung Smart Things

Was ist Samsung Smart Things?

Samsung Smart Things bezeichnet das Smart Home Ökosystem von Samsung. Die zentralen Elemente dabei sind der Samsung Smart Things Hub und die dazugehörige Smart Things App. Der Smart Things Hub stellt dabei die Steuerzentrale für alle angeschlossenen Komponenten dar. Der Hub wird mit dem Router verbunden und ermöglicht so eine Verbindung der Komponenten mit der Smart Things Cloud. Die Smart Things Cloud stellt dabei die Schnittstelle zwischen Diensten, der Smart Things App und den angebotenen Apps dar. Die Smart Things App ermöglicht das Überprüfen, Automatisieren und das Einstellen der Komponenten.

Überprivilegierung

Das Framework von Samsung Smart Things besitzt bereits Sicherheitsmechanismen, um ungewolltes Verhalten zu unterbinden. Beispielsweise werden Apps Privilegien, in diesem Fall „capabilities“ genannt, zugestanden und diese bestimmen, welche Operationen auf welchem Smart Device initiiert werden darf.

Das „principle of least privilege“ (POLP) ist weit verbreitet und besagt, dass die Zugriffsrechte so stark wie möglich eingeschränkt werden sollen, ohne den Arbeitsfluss zu stören. So wenig Rechte wie möglich, so viele wie nötig. In der Theorie sollte so sichergestellt sein, dass kein Nutzer oder Anbieter mehr Rechte bekommt, als ihm zustehen.

Samsung Smart Things setzt dieses Verfahren mittels einer Subscription von Smart Apps auf einem Event um. Will eine Application eine Berechtigung für ein Device bekommen, wird diese angefragt und muss vom Nutzer autorisiert werden.

Eine Subscription kann man sich wie ein Abonnement vorstellen. Wenn eine App eine Subscription für ein Event eines Smart Device abgeschlossen hat, würde diese App benachrichtigt werden, wenn so ein Event generiert wird.

In der durchgeführten Analyse wurde herausgefunden, dass 55% der Applications mehr Berechtigungen angefordert haben, als wirklich benutzt wurden. Jedoch haben auch 42% mehr Berechtigungen bekommen, als angefordert wurden.

Dies entsteht durch die Device Subscriptions. Wenn eine Application eine Berechtigung für ein Smart Device, zum Beispiel ein Türschloss, anfordert, werden dieser Application alle Berechtigungen für dieses Device eingerichtet. Wenn die App beispielsweise nur das Türschloss abschließen möchte, würde mit dieser Anfrage auch alle anderen Aktionen des Türschlosses ermöglicht werden. Bei einem Türschloss wäre dies auch ein „unlock“ Kommando.

Capability	**Commands**	**Attributes**
capability.lock	lock(), unlock()	lock(lock status)
capability.battery	N/A	battery(battery status)
capability.switch	on(), off()	switch(switch status)
capability.alarm	off(), strobe(), siren(), both()	alarm(alarm status)
capability.refresh	refresh()	N/A

Beispiele von Capabilities(Fähigkeiten) im SmartThings Framework[1]

Effektiv bedeutet dies, dass zwar nur die Berechtigungen angefordert wurden, die gebraucht werden, aber trotzdem mehr Berechtigungen gewährt werden. Dies widerspricht dem „principle of least privilege“, da es hierbei nun um eine Überprivilegierung handelt.

Event Spoofing

Events werden von Smart Devices genutzt um Informationen zu verbreiten. Wenn ein Gerät eine Änderung durchführt, die dem Hub mitgeteilt werden muss, erstellt das Device ein Event. Es ist möglich, dass diese Events über das Einklinken in den Benachrichtigungsfeed von anderen Geräten ausgelesen werden können.

Damit Events nur hervorgerufen werden dürfen, wenn der Ursprung über die nötige Autorisierung verfügt, müsste nachvollziehbar sein, welches Device ein Event auslöst. Um den Ursprung eines Events nachvollziehen zu können existieren drei Identifier. Der Location Identifier enthält die Information des Ortes, an dem das Event auftritt. Der Hub Identifier ist die Bezeichnung des Hubs, mit dem das Smart Device verbunden ist. Der Device Identifier ist eine ID, die der Hub einem Smart Device zuteilt, wenn dieses am Hub angemeldet wird.

Wenn ein smart Rauchmelder also Rauch entdeckt, wird ein „Rauch“-Event ausgelöst. Dieses Event enthält weitere Informationen, wie einen Location Identifier, einen Hub Identifier und einen Device Identifier.

Man braucht 3 Identifier, um ein Event vorzutäuschen. Der Location und Hub Identifier sind allen Smart Applications automatisch bekannt. Der Device Identifier hingegen ist zwar statisch im Smart Things Netzwerk, aber wird erst beim Hinzufügen des Gerätes in das Netzwerk generiert und zugewiesen. Eine Möglichkeit diesen Device Identifier herauszufinden, ist eine WebService SmartApp auszunutzen und diese dazu zu zwingen, eine Liste aller Device Identifier zu versenden. Hierüber können jedoch nur die Device Identifier herausgefunden werden, für welche die SmartApp auch autorisiert wurde.

Ein WebService ist ein Dienst, der per http oder https angesprochen werden kann. Meist sind dies Dienste, die zur Informationsgewinnung genutzt werden. Wird ein WebService angesprochen, so antwortet dieser mit den angeforderten Informationen.

Eine WebService SmartApp wäre also eine SmartApp, die über eine Netzwerkverbindung aufgerufen werden kann.

Wenn der Device Identifier bekannt ist, kann eine SmartApp Events vortäuschen und dadurch das Verhalten von anderen Applications, die auf Events basieren, auslösen. So könnte beispielsweise ein Urlaubsmodus, durch ein vorgetäuschtes Event deaktiviert werden und das Haus unsicherer machen.

Zusammenfassend ist das Event Subsystem sehr unsicher. Von 132 analysierten Geräten haben 111 Geräte ihren Status und andere sensible Daten häufig gepostet.

Third Party Integration

SmartApps können auch HTTP Endpunkte darstellen. Als WebService können sie auf GET, POST, PUT und DELETE Requests reagieren. Ein großer Anbieter solcher Services ist „If this then that“ [8]. Die Endpunkte der Kommunikation sind dabei mittels des OAuth Protokolls gesichert. „OAuth (Open Authorization [9]) ist ein offenes Protokoll, das eine standardisierte und sichere API-Autorisierung für Web-, Desktop- und Mobile-Anwendungen erlaubt.“(Wikipedia) Prinzipiell ist dies ein sicheres Protokoll, jedoch hängt diese Sicherheit von der vernünftigen Implementierung seitens eines Entwicklers ab. Hierbei hat sich gezeigt, dass dies nicht immer der Fall ist. Einige Applications, die im Google Play Store verfügbar sind, benutzen keine zusätzliche Autorisierungsschicht, sondern speichern die Anmeldedaten direkt im Bytecode.

Dies ermöglicht es, über einen Exploit an die Anmeldedaten zu gelangen und somit Zugriff auf geschützte Daten zu erhalten.

WebApp Input

Die Endpunkte der WebServices sind per OAuth geschützt, jedoch steht es den Entwicklern frei, die Art und Arbeitsweise der Endpunkte komplett zu bestimmen. Eine der Möglichkeiten wäre Groovy zu benutzen.

Groovy ermöglicht es Methoden dynamisch über ihren Namen aufzurufen. Unter Annahme eines Groovy Strings ‘def str = “foo“‘, ließe sich die dahinterstehende Methode „foo“ mit „$str“ aufrufen. Häufig wird dies genutzt, um dynamisch Methoden aufzurufen, in dem der String der auszuführenden Methode per HTTP gesendet wird.

Applications, die auf diese Art des Methodenaufrufs setzen, sind jedoch anfällig für Angriffe, welche die Überprivilegierung ausnutzen und so Aktionen initiieren, die für diese Application eigentlich nicht ausführbar sein sollten. Ein sehr ähnlicher und bekannter Exploit dieser Art, wären SQL-Injections.

External Communication API

SmartThings benutzt das OAuth Protokoll um eingehende Requests aus dem Internet zu authentisieren. Jedoch gibt es keine Auflagen für ausgehende Requests von SmartApps. Ebenso bietet SmartThings selbst einen Service an, um SMS versenden zu können.

Hierdurch können vertrauliche Daten per SMS oder E-Mail versendet werden.

Auswirkungen

Da SmartApps nicht lokal auf dem SmartHub, sondern nur in einer Cloud ausgeführt werden, ist es nicht ohne Weiteres möglich den Bytecode zu analysieren. Jedoch existiert eine Web IDE, auf der Entwickler ihre Groovy-Programme auf Quellcode-Ebene austauschen können. Über diesen Umweg wurden insgesamt 499 SmartApps heruntergeladen, um sie analysieren zu können.

TABLE II
BREAKDOWN OF OUR SMARTAPP AND SMARTDEVICE DATASET

Total # of SmartDevices	**132**
# of device handlers raising events using `createEvent` and `sendEvent`. Such events can be snooped on by SmartApps.	111
Total # of SmartApps	**499**
# of apps using potentially unsafe Groovy dynamic method invocation.	26
# of OAuth-enabled apps, whose security depends on correct implementation of the OAuth protocol.	27
# of apps using unrestricted SMS APIs.	131
# of apps using unrestricted Internet APIs.	36

Abbildung 2: Zusammengefasste Auswertung der SmartApps [1]

Überprivilegierung

TABLE IV
OVERPRIVILEGE ANALYSIS SUMMARY

Reason for Overprivilege	# of Apps
Coarse-grained capability	276 (55%)
Coarse SmartApp-SmartDevice binding	213 (43%)

Abbildung 3: Analyse der Überprivilegierung [1]

Um die Überprivilegierung bewerten zu können, wurden 499 SmartApps ausgewählt und untersucht. Hierbei wurde die Menge an angeforderten Berechtigungen mit der Menge an tatsächlich erhaltenen Berechtigungen verglichen. In 276 Fällen hatte die Application mehr Berechtigungen als sie angefordert hatte. Hierbei handelt es sich um Überprivilegierung, die entsteht wenn eine SmartApp eine gewisse Berechtigung direkt bei einem Gerät anfordert.

Wenn eine SmartApp über die Präferenzen bestimmte Berechtigungen erhalten möchte, wird sie von dem Benutzer für bestimmte Geräte autorisiert. Bei der Untersuchung dieser Art der Überprivilegierung wurde festgestellt, dass 213 von den untersuchten Applications mehr Berechtigungen erhielten, als ursprünglich angefordert wurden.

Beispiele, die dies ausnutzen

Einige SmartApps nutzen dieses Verhalten aus, um zusätzliche Funktionen bereit zu stellen. Zwei Beispiele hierfür sind die „Gentle Wake Up“ und „Welcome Home Notification“ Applications.

„Gentle Wake Up“ ist eine App, die langsam die Helligkeit des Lichtes erhöht, um so einen angenehmen Aufwachvorgang zu ermöglichen. Wenn die Lampe dies unterstützt, wird auch die Farbe des Lichtes angepasst. Das Anpassen der Farbe erfordert jedoch Berechtigungen, die die App nie angefordert hat.

Die „Welcome Home Notification“ App benutzt einen Sonos Lautsprecher, um ein Musikstück abzuspielen, wenn eine Tür geöffnet wird. Jedoch wird neben der Berechtigung für den „musicPlayer“ auch die Berechtigung benutzt, um den Lautsprecher an- und auszuschalten. Dies ist jedoch eine Berechtigung, die nie angefordert wurde.

In beiden Fällen nehmen die Entwickler an, dass es der App gestattet ist, dies auszuführen, obwohl sie dies nie explizit angefordert haben.

Dies zeigt, dass dieser Fehler bekannt ist und ausgenutzt wird.

Im weiteren Verlauf werden 4 Möglichkeiten gezeigt, wie die bereits aufgeführten Sicherheitslücken ausgenutzt werden können.

TABLE V
FOUR PROOF-OF-CONCEPT ATTACKS ON SMARTTHINGS

Attack Description	Attack Vectors	Physical World Impact (Denning *et al.* Classification [12])
Backdoor Pin Code Injection Attack	Command injection to an existing WebService SmartApp; Overprivilege using SmartApp-SmartDevice coarse-binding; Stealing an OAuth token using the hard-coded secret in the existing binary; Getting a victim to click on a link pointing to the SmartThings Web site	Enabling physical entry; Physical theft
Door Lock Pin Code Snooping Attack	Stealthy attack app that *only* requests the capability to monitor battery levels of connected devices and getting a victim to install the attack app; Eavesdropping of events data; Overprivilege using SmartApp-SmartDevice coarse-binding; Leaking sensitive data using unrestricted SMS services	Enabling physical entry; Physical theft
Disabling Vacation Mode Attack	Attack app with no specific capabilities; Getting a victim to install the attack app; Misusing logic of a benign SmartApp; Event spoofing	Physical theft; Vandalism
Fake Alarm Attack	Attack app with no specific capabilities; Getting a victim to install the attack app; Spoofing physical device Events; Controlling devices without gaining appropriate capability; Misusing logic of benign SmartApp	Misinformation; Annoyance

Bild 4: 4 Beispiele für Angriffe auf Samsung Smart Things [1]

Die 4 Möglichkeiten sind ein Backdoor-Angriff, um die Zugriffsdaten für ein Türschloss zu bekommen, ein Angriff, bei dem die Zugriffsdaten des Türschlosses „erschnüffelt“ werden, eine Möglichkeit, den Ferien-Modus auszuschalten, indem ein Event vorgetäuscht wird und ein Angriff, bei dem ein Alarm fälschlicherweise aktiviert wird.

Der Backdoor Angriff lässt sich in zwei Teile aufteilen. Zuerst wird der OAuth Token ermittelt und danach mittels Groovy Injection versucht ein SmartDevice zu kontrollieren.

Das OAuth Token lässt sich anfordern, wenn die Client ID und eine Passphrase bekannt sind. Dies kann erreicht werden, in dem der Benutzer für die Autorisierung seine Accountdaten zwar auf der echten SmartThings Seite eingibt, die Daten jedoch über den „redirect“-Part der Url an eine Zwischenstelle weitergeleitet werden. Mit der ClientID, der Passphrase und dem geheimen Passwort, welches aus dem Bytecode bestimmter Apps ausgelesen werden kann, ist es beispielsweise einen eigenen Schlüssel für das Schloss zu setzen. SmartThings erlaubt dem Besitzer des Tokens die zugehörigen Aktionen auf dem SmartDevice auszuführen.

Wenn nun ausgenutzt wird, dass per dynamischem Groovy Methodenaufruf, eine eigens geschriebene Methode aufgerufen werden kann, kann beispielsweise der Schlüssel für das Schloss verändert werden. Dies könnte genutzt werden, um Zugang zu einem Haus zu erlangen und würde folglich den eigentlichen Sinn des Schlosses außer Kraft setzen.

Das Opfer muss sich in diesem Fall nur einmal auf der eigentlich richtigen SmartThings Website anmelden. Die Anmeldeinformationen werden jedoch vom Angreifer abgefangen. Voraussetzung für das Kontrollieren des Schlosses ist jetzt nur das Vorhandensein einer App, die das geheime Passwort im Bytecode speichert. Diese Information könnte vom Angreifer ausgelesen werden und damit hätte der Angreifer alle Informationen, die er braucht, um SmartThings vorzugaukeln, dass er die Rechte hat beispielsweise den Schlossstatus zu ändern.

Der zweite Angriff benutzt eine App, die den Batteriestand von Smart Devices abfragen kann, jedoch im Hintergrund die Sperrcodes für ein Türschloss ausspioniert.

Die App fragt ordnungsgemäß nur nach den Berechtigungen, um auf den „capability.battery“ Befehl zugreifen zu dürfen. Wenn der Sperrcode des Türschlosses verändert wird, teilt das Schloss diese Änderung dem Hub mit. Hierbei wird ein „codeReport“ Event generiert. Wenn die App sich für die „codeReport“ Events registriert, kann die App die Daten, die das Event mit sich bringt, auslesen.

```
zw device:02,
command:9881,
payload:00 63 03 04 01 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
parsed to
[['name':'codeReport', 'value':4,
'data':['code':'8877'],
'descriptionText':'ZWave Schlage Lock code 4 set',
'displayed':true,
'isStateChange':true,
'linkText':'ZWave Schlage Lock']]
```

Listing 3. Sample codeReport event raised when a code is programmed into a ZWave lock.

Abbildung 5: Code Beispiel: Codereport bei einem ZWave Schloss [1]

Wie auf dem Bild deutlich wird, steht der neue Code im Klartext in diesem Event. Die App kann den neuen Sperrcode für die Tür also quasi mitlesen. Da der SmartThings Hub nur Restriktionen auf eingehenden Nachrichten hat, wäre es nun kein Problem, diesen Türcode mittels einer SMS an einen potenziellen Angreifer weiterzuleiten.

Es ist es sehr leichtsinnig Informationen, wie einen Sperrcode, unverschlüsselt zu versenden. Umso mehr, wenn alle für das Gerät autorisierten Applications diese Informationen einsehen können.

Der zweite Fehler ist die Möglichkeit, der „dynamic Method Invocation“. In diesem Fall wurde diese genutzt, um eine SMS abzuschicken. Diese Nachricht verließ ohne Einschränkungen das Haus.

Effektiv könnte eine Familie so komplett ausspioniert werden.

Eine andere Möglichkeit, die Schwachstellen im Eventsystem des Samsung SmartThings Hub auszunutzen, ist falsche Events zu simulieren. Potenziell kann jede App ein Event simulieren, welches ein bestimmtes Verhalten provozieren kann.

Ein Beispiel hierfür ist die Möglichkeit ein Haus trotz Abwesenheit bewohnt aussehen zu lassen; häufig auch „Vacation-Mode“ genannt.

Mittels einer App könnte ein solcher Modus zurückgesetzt und so effektiv abgeschaltet werden. Wenn dieser Modus zusätzlichen Schutz wie beispielsweise Kameras mit einschaltet, würde dies bedeuten, dass das Haus nicht nur ungeschützt aussieht, sondern auch ungeschützt wäre.

Sehr ähnlich ist das Simulieren eines physischen Events. Hierüber könnte eine Application einem Rauchmelder vorgaukeln, es gäbe eine Rauchentwicklung, woraufhin dieser natürlich den Bewohner wecken bzw. auf sich aufmerksam machen möchte. In diesem Fall eines Rauchmelders ist dies primär unangenehm und könnte für Schlafmangel bei dem betroffenen Individuum führen, jedoch ergibt sich hieraus keine direkte Möglichkeit das Haus zu betreten.

Jedoch ist es denkbar, dass in einem weiter vernetzten Haus ein möglicher Feueralarm dafür sorgen könnte, dass sich Fluchtwege von selbst öffnen.

Zusammenfassend lässt sich feststellen, dass die Angriffsmöglichkeiten sehr vielfältig sind. Die Voraussetzungen für die gezeigten Angriffe sind jedoch das Installieren einer Application mit schadhaftem Inhalt.

Den Benutzer zu überzeugen, eine infizierte oder schadhafte Application zu installieren, ist also die größte Hürde. Die hier angesprochene Battery-Monitor App ist jedoch ein Beispiel für eine scheinbar einfache Utility-App, die allerdings durch die Überprivilegierung, das offene Event-System und die dynamische Methodeninvokation sehr mächtig werden kann.

Besonders die Möglichkeit über einen dynamischen Methodenaufruf, eine eigens gebaute Methode aufzurufen kann sehr gefährlich werden. Effektiv gibt es nur die Hardware Beschränkungen des Devices, welches die Möglichkeiten der Methoden einschränkt.

Schutzmöglichkeiten

Da drei der vier Angriffsmöglichkeiten das Installieren einer bestimmten App voraussetzen, die entweder infiziert bzw. schadhaft ist, oder bestimmte Sicherheitsrichtlinien nicht ordentlich implementiert hat, bleibt die Frage wie man sich davor schützen kann.

Natürlich ist als erstes Vorsicht geboten, wenn man Apps installiert. Hierbei sollte überlegt werden, ob z.B. eine Batterie-App wirklich von Nöten ist, oder ob man diese doch nicht braucht.

Bei Sicherheitsmaßnahmen wie Kameras bietet es sich an, diese nur angeschlossen und eingerichtet zu haben, wenn man diese auch benutzen würde. Wenn man seinen normalen Alltag hat und die Kamera nicht benutzt, kann diese auch vom Strom getrennt sein und nur, um nach Updates zu suchen, regelmäßig verbunden werden. Verlässt man hingegen das Haus für einen Urlaub und plant diese Kamera zu benutzen, könnte diese wieder angeschlossen werden. So könnte das Risiko des ausspioniert werden, umgangen werden.

Da es sich bei den meisten Problemen, um interne Probleme des Hubs oder von Applications handelt, kann der Endverbraucher hier leider nicht viel tun, um sich selbst zu schützen.

Risiken

Wenn es für den Bewohner eines Smart Home nicht möglich ist, die Sicherheitslücken selbst zu minimieren stellt sich die Frage, welchen Risiken er sich durch das Benutzen dieses Hubs aussetzt.

Da die Smart Devices von außerhalb gesteuert, beziehungsweise beeinträchtigt werden können, ist unter Umständen die Funktionsweise der Geräte nicht mehr sichergestellt. Ein Türschloss, welches ungebetene Gäste nicht aufhält, widerspricht dem eigentlichen Sinn eines Schlosses.

Wenn jedoch auch Kameras oder Mikrofone im Smart Home mit eingebunden sind, ergibt sich zusätzlich das Risiko, dass Kamera und Mikrofon genutzt werden, um das Haus auszuspionieren. Über eine Kamera könnten tägliche Routinen erfasst und analysiert werden, um herausfinden zu können, wann ein Haushalt unbewohnt ist. Dabei könnte das Mikrofon benutzt werden, um vorher gefundene Informationen zu verifizieren. Ebenso könnte jedoch auch versucht werden, die gefundenen Audiodateien zu analysieren, um mögliche Informationen über zukünftige Pläne der Bewohner herauszufinden. Dies würde jedoch einen erheblichen zusätzlichen Aufwand mit sich ziehen.

Neben der Beeinträchtigung der Privatsphäre, die sich aus der Beobachtung durch Kamera und Mikrofon ergeben kann, ist auch ein Einbruch eine mögliche Folge des Ausspionierens. Wenn genug Informationen gesammelt wurden, ließe sich daraus berechnen, wann der Haushalt in der Regel unbeaufsichtigt ist. Da Kontrolle über die Kameras vorhanden ist, wäre es dementsprechend möglich diese einfach auszuschalten.

Ein anderes Risiko wurde bereits aufgezeigt und bezieht sich auf das Ausspionieren der Sperrcodes zu einem Smart Lock. Ist dieser „Schlüssel“ bekannt, so besteht ebenfalls die Gefahr eines Einbruches.

Die größte Gefahr stellen jedoch die dynamischen Methodenaufrufe dar. Durch diese kann eine sogenannte „arbitrary code execution“ provoziert werden. Hierbei wird Code ausgeführt, der erst durch diesen Angriff in das System kam. Die limitierenden Faktoren wären in diesem Fall nur die Leistungsfähigkeit des Gerätes auf dem der Code ausgeführt wird und die möglichen Restriktionen, die die Umgebung dem Programm auflegt.

Arbitrary Code Execution bezeichnet das Ausführen von fremdem Code auf einem Gerät. Hierdurch kann ein Angreifer seinen eigenen Programmcode auf eine Maschine schicken und ihn dort ausführen. Der Angreifer kann also komplette Kontrolle über die Tätigkeiten des Gerätes erlangen, während das Opfer dies nicht mitbekommt.

Risikobewusstsein

Natürlich sind dies nur mögliche Risiken, die nicht eintreffen müssen. Jedoch sollte sich jeder Besitzer von Smart Devices über diese Möglichkeiten Gedanken machen. Diesbezüglich wurden wieder 1025 Personen, über bestimmte Risiken im Bezug auf Smart Home Devices befragt. Diese 1025 Personen sind alles Personen gewesen, die bereits Erfahrungen mit Smart Home Devices gesammelt haben.

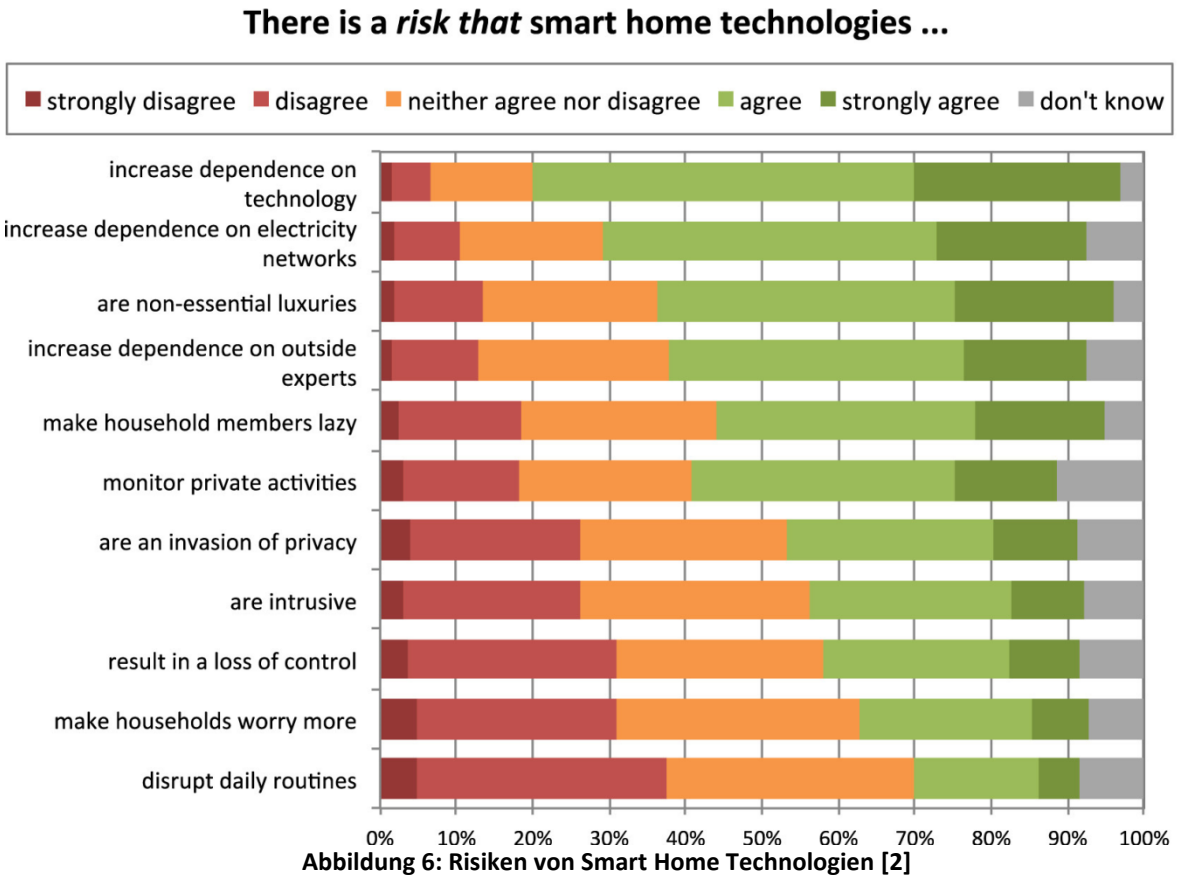

Abbildung 6: Risiken von Smart Home Technologien [2]

Über die Hälfte der Befragten sind der Meinung, dass ein hohes Risiko existiert, dass Smart Home Technologien Luxusartikel sind, die nicht zwingend notwendig sind. Dies könnte ein Risiko darstellen, da so eine unnötige technische Abhängigkeit in die eigenen 4 Wände gelangt. Mehr Technik bedeutet häufig auch mehr Möglichkeiten angegriffen zu werden.

Da die Erhöhung des Wohnkomforts einer der Hauptvorteile ist, der in Smart Home Technologien gesehen wird, ist es nachvollziehbar, dass diese Technologie als nicht essentiell angesehen wird.

Mehr als 70% sind der Meinung, dass Smart Home Technologien sowohl die Abhängigkeit von den Technologien, als auch die Abhängigkeit von den elektrischen Netzwerken erhöhen.

Je mehr Smart Devices in einen Wohnraum integriert werden, desto stärker wird sich darauf verlassen, dass diese Geräte so funktionieren, wie es beabsichtigt ist. Je kritischer der Punkt ist, an dem das Gerät installiert wird, desto relevanter ist es, dass alles reibungslos funktioniert.

Weitere relevante Punkte sind die Beeinträchtigung der Privatsphäre, der Kontrollverlust, die Tatsache, dass man sich mehr Sorgen macht und die Beobachtung privater Aktivitäten.

Interessanterweise unterscheiden sich die Punkte „Beeinträchtigung der Privatsphäre“ und „Beobachtung privater Aktivitäten“ um knapp 10 Prozentpunkte. Ein möglicher Schluss hieraus ist, dass einige Personen sich zwar beobachtet fühlen, dies aber nicht als ein potenzielles Risiko ansehen. Unterstützt wird dieser Schluss dadurch, dass nur ca. 30% der Befragten angegeben haben, dass Smart Home Technologien Grund für mehr Beunruhigung sind. Hier wird der Vorteil stärker gewichtet, als mögliche Nachteile.

Aktueller Stand

Natürlich wurden die Entwickler der benutzten Applications und die zugehörige Stelle für Samsung SmartThings über die gefundenen Schwachstellen informiert. Die Entwickler der SmartApps haben angegeben, dass sie mit SmartThings zusammenarbeiten, um die Probleme zu beheben.

Samsung SmartThings wiederum hat eine dedizierte Abteilung, die die Apps aus dem SmartThings-Store prüft. Im Bezug auf das dynamische Aufrufen von Methoden versucht das SmartThings-Team einerseits in den Richtlinien das Verwenden von dynamischer Methodeninvokation als „Bad Practice“ darzustellen und andererseits werden http Endpoints in den Applications von dem Team untersucht. [10]

Bezüglich des „principle of least privilege“ ist die Intention, den Anwendungsbereich der möglichen Aktionen so gering wie möglich zu halten. Dies deutet darauf hin, dass Einschränkungen in der Berechtigungsvergabe vorgenommen werden sollen. Effektiv wird dadurch eine strengere Berechtigungsvergabe angestrebt, die eine Überprivilegierung, wie sie bisher vorhanden war, verhindern soll.

Bei einigen Punkten ist es leider nicht möglich nachzuvollziehen, inwiefern diese umgesetzt wurden.

Zusammenfassung

Zusammenfassend lässt sich sagen, dass die Sicherheitslücken, die in den Systemen des Samsung Smart Things Hub gefunden wurden, sehr weitgreifend und tiefläufig waren. Hierbei war es möglich durch gezieltes Ausnutzen der Lücken nahezu alles manipulieren zu können.

Nachdem Samsung diese Lücken mitgeteilt wurden, waren sie in der Pflicht, diese weitestgehend zu schließen. Leider wurden keine Informationen gefunden, um das Schließen dieser Lücken zu verifizieren. Von daher wird im Weiteren angenommen, dass das Smart Things Security Team dies, wie angekündigt, umgesetzt hat.

Fazit

Smart Home Technologien sollen primär den Komfort erhöhen. Jedoch gibt es auch viele Hersteller, die Devices entwickeln, die für Sicherheitszwecke benutzt werden. Die größten Beispiele hierfür wären Türschlösser und Kameras. Im Gegensatz zu den non-Smart Türschlössern gibt es bei den smarten den Nachteil, dass sie eine Verbindung zum Internet benötigen. Wenn für ein normales Schloss ein weiterer Schlüssel gebaut werden soll, ist die physische Anwesenheit ein entscheidender Faktor. Bei Smart Devices ist jedoch genau diese Anwesenheit nicht mehr notwendig. Wie an den Beispielen gezeigt wurde, war es theoretisch gesehen möglich diese Schlüssel zu bauen, ohne Anwesenheit zeigen zu müssen. Die einzige Voraussetzung war, dass in dem Haushalt eine bestimmte App heruntergeladen werden musste.

Genau deswegen wäre es hier notwendig, dass das Sicherheitssystem perfekt funktioniert. Leider war genau dies beim Smart Things Hub nicht der Fall.

Aus diesen Gründen halte ich den Security Teil der Smart Home Devices noch nicht für ausgereift genug. Geräte wie Smart Bulbs, bei denen im Notfall der Strom abgestellt werden kann, sind hiervon nicht so stark betroffen, wie Kameras oder Türschlösser, die im schlimmsten Fall einen einfachen Zugang zum Haus ermöglichen.

Effektiv muss jede Person die Gefahren und Vorteile abwägen und eine eigene Entscheidung treffen. Man sollte sich bewusst sein, dass diese Risiken bestehen und dass theoretisch gesehen jeder Haushalt ein mögliches Ziel ist.

Literaturverzeichnis:

[1] Earlence Fernandes, Jaeyeon Jung, Atul Prakash
„Security Analysis of Emerging Smart Home Applications“
2016 IEEE Symposium on Security and Privacy
http://ieeexplore.ieee.org/document/7546527/
© 2011 IEEE
[abgerufen am 01.11.2017]

[2] Charlie Wilson, Tom Hargreaves, Richard Hauxwell-Baldwin
”Benefits and risk of Smart home technologies”
Energy Policy, April 2017 Pages 72-83
https://www.sciencedirect.com/science/article/pii/S030142151630711X
lizensiert unter http://creativecommons.org/licenses/by/4.0/
[abgerufen am 01.11.2017]

[3] TechTarget: TechTarget Definitionen
URL:http://searchsecurity.techtarget.com/definition/principle-of-least-privilege-POLP, o.J.
[abgerufen am 15.04.2018]

[4] SmartThings: Samsung Smart Things Dokumentation
URL:http://docs.smartthings.com/en/latest/latest-updates.html#march-22-2017, o.J.
[abgerufen am 06.04.2018]

[5] Amazon.com: Amazon Alexa
URL: https://developer.amazon.com/de/alexa, o.J.
[abgerufen am 06.04.2018]

[6] Google: Google Home
URL:https://store.google.com/product/google_home, o.J.
[abgerufen am 06.04.2018]

[7] Samsung: Samsung Smart Things
URL:https://www.samsung.com/us/smart-home/smartthings/, o.J.
[abgerufen am 06.04.2018]

[8] IFTTT Inc.: „If this then that“
URL:https://ifttt.com/, o.J.
[abgerufen am 06.04.2018]

[9] Aaron Parecki, Chris Messina: Open Authorization
URL:https://oauth.net/, o.J.
[abgerufen am 06.04.2018]

[10] SmartThings: Samsung Smart Things Code Review Guidelines
URL: http://docs.smartthings.com/en/latest/code-review-guidelines.html, o.J.
[abgerufen am 15.05.2018]