Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Thomas Jäger

Security Information and Event Management im Smart Home

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Thomas Jäger

Security Information and Event Management im Smart Home

Masterarbeit eingereicht im Rahmen der Masterprüfung

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Bettina Buth
Zweitgutachter: Prof. Dr. Klaus-Peter Kossakowski

Eingereicht am: 27.02.2018

Thomas Jäger

Thema der Arbeit
Security Information and Event Management im Smart Home

Stichworte
Security, Smart Home, SIEM, Detect, IoT, Use Case, Kill Chain, Korrelation, ZigBee, OpenHAB

Kurzzusammenfassung
Inhalt dieser Arbeit ist die Untersuchung der Übertragbarkeit des SIEM Ansatzes zur Angriffserkennung auf das Smart Home. Dabei wird die Notwendigkeit dieser Übertragung gezeigt und eine Herangehensweise erarbeitet. Anhand mehrerer ausgewählter Angriffsszenarien und einer aufgesetzten Smart Home Testumgebung wird ein SIEM-System aus Open Source Komponenten implementiert. Durch Nachstellen der Angriffsszenarien wird die Funktionalität des entwickelten Systems getestet.

Thomas Jäger

Title of the thesis
Security Information and Event Management in the Smart Home

Keywords
Security, Smart Home, SIEM, Detect, IoT, Use Case, Kill Chain, Correlation, ZigBee, OpenHAB

Abstract
This thesis examines the transferability of the SIEM approach for detecting attacks to Smart Home environments. For that the need of this transfer is shown and an own approach is developed. Based on multiple choosen attacks and a set up exemplary Smart Home environment, a SIEM-System is build from Open Source Components. By emulating the attack scenarios, the functionality of the developed system is tested.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die Geschwindigkeit, in der die Entwicklung des Internet of Things (IoT) voranschreitet und vorangetrieben wird, ist rasant und nimmt stetig zu. Dabei ist sowohl die quantitative Expansion als auch die qualitative Komponente der Weiterentwicklung zu betrachten.
Quantitativ wächst die Zahl der IoT-Geräte stark an. Während eine von Gartner erhobene Statistik aus dem Jahr 2016, zu sehen auf Abbildung 1.1, für das Jahr 2020 eine Gesamtzahl von 20,4 Milliarden Geräten prognostiziert und die International Data Cooperation (IDC) für das selbe Jahr auf eine Zahl von 30,1 Milliarden zum Internet verbundene IoT-Geräten kommt [MacGillivray, 2013], schätzte Cisco diese Anzahl für das Jahr 2025 bereits auf bis zu 50 Milliarden Geräte [Manyika u. a., 2015]. Auch die erwarteten wirtschaftlichen Kennzahlen sind bemerkenswert. Eine ausführliche Studie von McKinsey aus dem Jahr 2015 erwartet einen wirtschaftlichen Mehrwert von 11 Billionen US Dollar durch das Internet of Things. Davon sollen 300 Millionen US Dollar auf den Bereich Smart Home fallen [Manyika u. a., 2015].

Doch der Fortschritt in diesen Bereichen kann nicht nur durch faktische Zahlen und Prognosen gemessen werden. Hinzu kommt der qualitative Bereich, der die Tatsache beschreibt, dass IoT-Geräte immer mehr Bereiche des Industriellen, Unternehmerischen, Medizinischen, Staatlichen und Privaten erobern. Die Industrie setzt IoT-Technologien beispielsweise in der Logistik und im Transportwesen ein, um Produktionsumgebungen oder Warenlager zu überwachen, indem Daten durch Sensoren gesammelt und ausgewertet werden. Im medizinischen Bereich werden IoT-Sensoren bspw. bereits zur Überwachung von Vitalwerten im alltäglichen Leben der Patienten verwendet [Atzori u. a., 2010]. Im Kampf gegen Luftverschmutzung findet das IoT seine Aufgabe in der Beschaffung von Messwerten [Suleman, 2017]. Längst wurden auch in eine Vielzahl von Autos IoT-Geräte integriert [Meola, 2016]. Sogar die im Jahr 2010 von Luigi Atzori et al. [Atzori u. a., 2010] noch unter der Kategorie 'Futuristic' aufgeführten 'Robot Taxis' klingen im Angesicht des bereits geänderten Gesetzes im Bundestag zum Autonomen Fahren [Bundestag, 2017], aber auch durch Entwicklungsvorhaben verschiedener Hersteller [Handelsblatt] mittlerweile gar nicht mehr nach Zukunftsmusik.
Ein besonders privater Bereich, der zunehmend durch das IoT durchdrungen wird, ist der

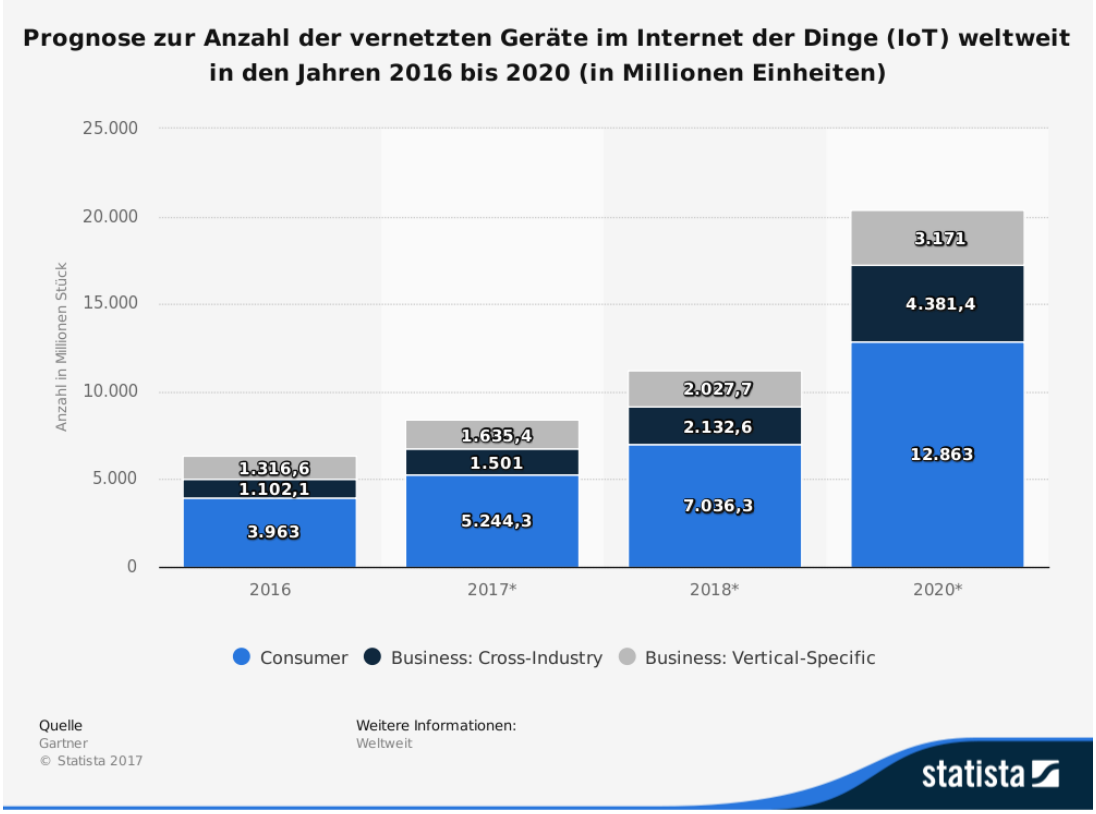


Abbildung 1.1: Prognose zur Anzahl der vernetzten Geräte im Internet der Dinge (IoT) weltweit in den Jahren 2016 bis 2020 (in Millionen Einheiten) [Statista, 2017]

Bereich des Wohnens. Als Smart Home werden grundsätzlich Wohnungen oder Häuser bezeichnet, deren elektronische Geräte und Dienste mithilfe eines Kommunikationsnetzes verbunden sind, gesteuert und überwacht werden können [UKEssays, 2013]. Zu diesen Geräten zählen unter anderem Türschlösser, Bewegungs- und Rauchmelder, Beleuchtungssysteme und Kameras [Sivaraman u. a., 2016]. Als Visionen hinter dem Begriff Smart Home sind neben dem gesteigerten Komfort unter anderem ein verbessertes altersgerechtes Wohnen [Kon u. a., 2017], eine Steigerung der Energieeffizienz [Allerding und Schmeck, 2011] und verbesserte Sicherheit gegenüber Wohnungseinbrüchen [Kodali u. a., 2016] zu finden. Laut einer Umfrage aus dem Jahr 2014 waren bereits 51% der amerikanischen Bürger bereit 500 $ für Smart Home Technologien auszugeben, um oben genannte Ziele zu erreichen [Sivaraman u. a., 2016]. Auch ist der Begriff des Smart Home laut einer Studie von Bitdefender aus dem Jahr 2016 unter den Bevölkerungen der USA, Frankreichs, Großbritanniens, Rumäniens, Australiens und Deutschlands verbreiteter als der Begriff des Internet of Things [Gheorghe, 2016].

Doch ausgerechnet der Wunsch nach mehr Sicherheit für die eigenen vier Wände und die daraus folgende Anschaffung von Smart Home Komponenten könnte sich destruktiv auf die Sicherheit der Bewohner auswirken. Der Entwicklung der Vernetzung und Eroberung des Alltags durch das IoT (aber auch durch andere IT-Technologien) steht ein starker Anstieg der Bedrohungslage durch IT-Angriffe gegenüber - mehr noch, sie geht damit einher. Im IoT-Umfeld sind beispielhaft die aufsehenerregenden Fälle wie die massiven DDoS-Attacken anzuführen, welche bis zu 145 000 aufgrund mangelhafter Sicherung kompromittierte IoT-Geräte als Botnetz nutzten [enisa, 2016]. Im Umfeld von Unternehmen und großen Infrastrukturen haben die weitreichenden Ransomware Angriffe wie WannaCry und Petya 2016 und 2017 großen Schaden angerichtet [O'Brien, 2017].
Auch wissenschaftliche Arbeiten und Versuche im Bereich der Smart Home Security haben gravierende Sicherheitslücken und daraus ermöglichte Angriffsformen skizziert, die bspw. zu einer unerwünschten Fernsteuerung von Smart Home Geräten führen [Notra u. a., 2014], diese zu einem Botnetz verbinden [Sivaraman u. a., 2016] oder Türschlösser unter fremde Kontrolle bringen [Rose und Ramsey, 2016] können. Auch ein aktuelles Projekt mit 3800 Zugriffsversuchen auf ein testweise aufgebautes Smart Home (mit ansteigender Prognose) zeigt die Relevanz der Thematik [Koramis, 2017].
Die drohende Gefahr durch schlecht gesicherte Smart Home Geräte scheint mittlerweile im Bewusstsein von Bevölkerungen angekommen zu sein. So sind laut der bereits zitierten Bitdefender Studie in erster Linie Bedenken im Bereich des Datenschutzes, gefolgt von der Angst, die Kontrolle über das eigene Smart Home zu verlieren oder ausgeraubt zu werden, verbreitet.
Die aufgeführten Angriffsformen sind unter anderem aufgrund ihrer Charakteristik als Zero-Day-Exploits oder Advanced Persistent Threats (APT)[1] erfolgreich [Thakar und Parekh, 2016].
Erkennend, dass sich hier Schwächen der herkömmlichen Schutzmechanismen wie Firewalls, Intrusion Prevention Systemen und Antivirenscannern, welche auf das Abwehren (Protect) von Malware oder bekannten Angriffsmustern ausgelegt sind, deutlich werden, verschiebt sich im Unternehmensumfeld der Security Fokus auf das Erkennen von bereits erfolgten oder sich in der Vorbereitungsphase befindlichen Angriffen (Detect). Erkennbar wird diese Verschiebung des Schwerpunktes im praktizierten Protect-Detect-React Ansatz, zu sehen in einer Studie von IDC aus dem Jahr 2017, in welcher 77 % der befragten Unternehmer Detect-Maßnahmen als wichtig oder sehr wichtig einstufen, wohingegen der bisherige Schwerpunkt auf Protect Mechanismen lag [IDC, 2017]. Einer der Ansätze zum Ausbau von Detect Mechanismen ist die Einführung eines Security Information and Event Management Systems (SIEM). Ein SIEM-

[1] APT-Angriffe zeichnen sich durch ihren mehrstufigen, über einen längeren Zeitraum erstreckenden Angriffsverlauf aus. Sie verwenden bevorzugt noch unbekannte Schwachstellen [Jangla u. a., 2015].

System hat die Echtzeit-Analyse unterschiedlichster Log- und Netzwerkverkehrs-Daten zum Ziel, um aus einer Korrelation dieser Daten erfolgte oder erfolgende Angriffe ableiten zu können [BSI, 2013].
Aufgrund der zuvor erläuterten Entwicklungen und des daraus entstehenden Bedarfs wird im Rahmen dieser Arbeit der Frage nachgegangen, ob sich ein SIEM Ansatz zur Erkennung von Angriffen und Angriffsvorbereitungen durch Sammeln, Korrelieren und Analysieren von Log- und Netzwerkverkehrs-Daten aus dem Unternehmenskontext auf Smart Home Umgebungen übertragen lässt und damit einen Teil zur Verbesserung der IT-Sicherheit im Smart Home beitragen kann.

1.1 Motivation

Das eigene Zuhause war schon immer ein Zufluchtsort, ein Ort des Schutzes und der Rückzugsmöglichkeit vor dem Alltag und der Aufdringlichkeit der Außenwelt [Fox, 2016]. In einer hektischen, vernetzten und globalisierten Welt mehr denn je. Es ist daher nicht verwunderlich, dass das Bedürfnis nach Sicherheit im eigenen Heim dementsprechend stark ausgeprägt ist. Jetzt, da die Entwicklungen des Internet of Things im Begriff sind, auch diesen privaten und intimen Bereich des alltäglichen Lebens zu durchdringen, ist es umso wichtiger, die möglichen Gefahren, die von dieser Entwicklung ausgehen, zu erforschen und Schutzmaßnahmen zu entwerfen. Das wissenschaftliche Ergründen von Gefahren und möglichen Schutzmaßnahmen kann auch Motivation und Anregung für Unternehmen, in diesem Fall für Produzenten von Smart Home Komponenten, sein, Vorschläge zu adaptieren und ihre eigenen, derzeit zu einem großen Teil nur mangelhaft gesicherten Geräte [Notra u. a., 2014] nachhaltig zu verbessern.
Wenn der Fortschritt im Bereich des Smart Home eine insgesamt positive Entwicklung nehmen soll, muss auch das Vertrauen der Menschen in diese Technologien verdient werden und wachsen. Die aktuelle Skepsis, gesehen in der Bitdefender Studie [Gheorghe, 2016], ist berechtigt und muss durch weitere Aufklärung und verantwortungsvolle Entwicklungen abgebaut werden.
Nur so kann es gelingen, dass das Smart Home den Menschen, die in ihm wohnen, zu Vorteilen verhilft, ihnen einen angenehmeren Alltag durch höheren Komfort und eine vereinfachte Steuerung ihrer alltäglichen Geräte ermöglicht, die Sicherheit des eigenen Zuhauses vor bspw. Einbrüchen gesteigert wird und Visionen wie ein verbessertes altersgerechtes Wohnen [Kon u. a., 2017] oder erhöhte Energieeffizienz [Allerding und Schmeck, 2011] umgesetzt werden können und die daraus entstehenden Risiken und Gefahren am Ende nicht dominieren.
Es ist auch im Interesse der Anbieter von Smart Home Geräten selbst, die Skepsis gegenüber ihren Produkten abbauen zu können, um langfristig und nachhaltig erfolgreich zu sein.

Diese Arbeit soll einerseits einen Teil zur Aufklärung von Problematiken, Risiken und Gefahren beitragen, aber auch einen Ansatz für eine mögliche positive Entwicklung in der Sicherung von Smart Home Umgebungen leisten, damit in Zukunft auch das vernetzte Zuhause ein sicherer, vertrauenswürdiger Ort des Rückzugs sein kann.

1.2 Vorgehensweise & Ziel

Ziel dieser Arbeit ist es, die mögliche Übertragung des SIEM Ansatzes aus dem Kontext von Unternehmen und anderen kritischen Infrastrukturen auf die Smart Home Umgebung zu prüfen. Dabei soll dem Ansatz auf Basis einiger ausgewählter Angriffsformen, welche auf eine typische Smart Home Umgebung ausgeführt werden könnten, begegnet werden. Die Erkennung dieser ausgewählten Angriffsformen durch den SIEM Ansatz soll der Prüfparameter für eine Bewertung der Übertragbarkeit sein. Wichtig ist dabei, dass eine ganzheitliche Abdeckung des Smart Home nur dadurch erfolgt, dass alle grundlegenden technischen und architektonischen Besonderheiten einbezogen werden.
Um diese Prüfung durchführen zu können, wird eine Eigenimplementation eines SIEM-Systems, bestehend aus angepassten Open Source Komponenten und selbst verfasstem Programmcode, vorgenommen. Durch eine Nachstellung der ausgewählten Angriffsszenarien auf eine Smart Home Versuchsumgebung wird die Prüfung anschließend durchgeführt.
Der Fokus dieser Arbeit liegt dabei klar auf der technischen Umsetzung und Machbarkeit des SIEM Ansatzes. Die operativen Prozesse, welche für den Betrieb eines SIEM-Systems im herkömmlichen Kontext notwendig sind, und deren Übertragbarkeit werden lediglich am Rande betrachtet.

1.3 Gliederung

Der Aufbau dieser Arbeit wird im Folgenden beschrieben. Die Kapitel 2 und 3 dienen als Grundlagenkapitel. Dabei werden in Kapitel 2 SIEM-Systeme in den Protect-Detect-React Ansatz eingeordnet und in ihrer Funktionsweise im herkömmlichen Kontext erläutert. Auch auf den operativen Betrieb, welcher nur am Rande betrachtet wird, und auf generelle Schwierigkeiten eines SIEM-Systems wird dabei eingegangen.
Kapitel 3 soll Grundlagen des Smart Home vermitteln. Dazu wird zunächst auf das IoT mit seinen Charakteristiken und Architekturansätzen eingegangen, um das Gezeigte auf das Smart Home als ein Teilgebiet des IoT zu übertragen. Dabei soll eine architektonische Gliederung des Smart Home entstehen, sodass diese Arbeit der Anforderung einer ganzheitlichen Betrachtung

gerecht werden kann. Darauf basierend wird die Security im IoT und Smart Home erläutert, bevor eine erste theoretische Übertragung von SIEM-Systemen in das Smart Home gewagt wird.
Im Zuge dessen werden in Kapitel 4 bereits vorhandene Ansätze diskutiert und aus den Resultaten der eigene Ansatz für die Umsetzung des SIEM Ansatzes innerhalb dieser Arbeit entwickelt.
Kapitel 5 dient der Dokumentation des Aufbaus einer Smart Home Versuchsumgebung anhand derer der SIEM Ansatz implementiert und getestet werden soll.
Um die Ziele für das zu implementierende SIEM-System zu definieren, werden in Kapitel 6 unterschiedliche Angriffsszenarien ausgewählt und erläutert. Diese werden in Bezug auf die zuvor entstandene architektonische Gliederung ausgewählt, um alle grundlegenden Aspekte eines Smart Home abdecken zu können. Im Anschluss werden die Angriffsszenarien in Kill Chains aufgebrochen, um eine Basis für die konkrete Use Case Entwicklung des SIEM Ansatzes zu erhalten.
Die eigentliche Umsetzung des eigenen Ansatzes wird in Kapitel 7 dokumentiert, um die zuvor theoretisch entwickelten Use Cases in Kapitel 8 praktisch zu implementieren.
In Kapitel 9 wird das implementierte SIEM-System mitsamt der entwickelten Use Cases in Betrieb genommen und getestet.
Mit einem Fazit über die gewonnenen Erkenntnisse und Ergebnisse der Übertragung des SIEM Ansatzes in das Smart Home und einem Ausblick auf mögliche zukünftige Projekte schließt diese Arbeit ab.

2 SIEM als Detect Prozess

Die wachsende Bedrohungslage gegenüber IT-Infrastrukturen von Unternehmen, kritischen Infrastrukturen wie öffentliche Verkehrsmittel und staatlichen Institutionen, ist dabei, den Prozess zur Bekämpfung von IT-Security Gefahren zu verändern (siehe Kapitel 1). In Folge dessen entstehen Ansätze wie der des SIEM-Systems.
Dieses Kapitel widmet sich zunächst einem Überblick über den Protect-Detect-React Ansatz. Anschließend werden die Gründe für die notwendige höhere Aufmerksamkeit auf den Detect-Anteil gezeigt, um danach den SIEM Ansatz, dessen Grundidee, Funktionsweise, operative Prozesse und Schwierigkeiten darzustellen. Damit soll dieses Kapitel ein Basisverständnis für SIEM-Systeme schaffen, um im späteren Verlauf die Übertragung auf Smart Home Umgebungen betrachten zu können. Auch sollen die dabei gezeigten funktionalen Prozessschritte herkömmlicher SIEM-Systeme Orientierung für die Implementierung des Smart Home SIEM-Systems bieten (siehe Kapitel 7).

2.1 Protect-Detect-React

Unter dem Protect-Detect-React Ansatz (auch Prevent-Detect-Response) versteht man einen allgemeingültigen Security Lifecycle (siehe Abbildung 2.1), der das Ziel hat, Vertraulichkeit, Verfügbarkeit und Vollständigkeit von Informationen und Systemen zu erreichen. In diesen drei Phasen können sich verschiedenste Ansätze oder Strategien widerspiegeln und sich somit in einem gesamtheitlichen Ansatz vereinheitlichen lassen. Der Protect-Detect-React Ansatz versteht das Erreichen von Security in einer IT-Umgebung nicht als Aufgabe mit erreichbarem Ziel, sondern als stetig andauernden Prozess [LaPiedra, 2002]. Konkrete Aufgaben oder Strategien innerhalb der einzelnen Phasen sind dabei jeweils von den vorherigen Phasen abhängig und können sich dementsprechend von Situation zu Situation verändern. Technologien, die zur konkreten Umsetzung von Security Mechanismen eingesetzt werden, übernehmen oft unterschiedliche Aufgaben in einer oder mehrerer dieser Phasen, sodass sie selten auf nur eine Phase festgelegt sind.
Um den Ansatz besser zu verstehen und die Einordnung eines SIEM-Systems vornehmen zu können, ist es notwendig, die Phasen zunächst einzeln zu betrachten. Im Jahr 2014 hat

Gartner eine Erweiterung des Protect-Detect-React Ansatzes um die Phase Predict als die Verbindungsphase von React zu Protect vorgeschlagen [Davidson, 2017]. Da diese vierte Phase im SIEM-Kontext von Belang sein kann, wird sie im Folgenden ebenfalls betrachtet.

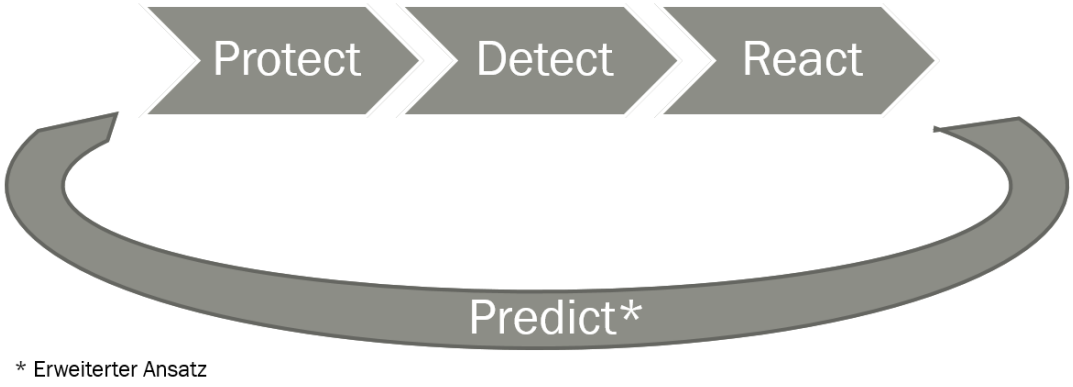


Abbildung 2.1: Protect-Detect-React Lifecycle [nach [Davidson, 2017]]

Protect

In der Protect Phase wird versucht, Gefahren und Angriffe zu verhindern oder zu unterbrechen bevor sie weiteren Schaden anrichten können. Dabei spielt die Architektur der IT-Infrastruktur eine große Rolle. Security Mechanismen wie die Zugangskontrolle, inklusive Authentifizierung, Autorisierung und Identifizierung, die nach definierten Richtlinien und Strategien aufgebaut ist, haben hier eine ebenso große Bedeutung wie das Schaffen von Bewusstsein für Gefahren, um bspw. Social Engineering[1] zu verhindern [LaPiedra, 2002]. Typische Technologien, die dieser Phase zugeordnet werden sind Anti-Malware/Antivirus-Lösungen, Firewalls und Intrusion Prevention Systeme (IPS), da diese Gefahren blockieren und verhindern, bevor sie Schaden anrichten können. Das kann allerdings missverständlich sein, denn dazu muss ergänzt werden, dass viele dieser Technologien ebenfalls eine Detect und React Phase durchleben, bevor sie, bspw. durch das Löschen von erkannter Malware, ihre Protect Aufgabe erfüllen [Davidson, 2017].
Charakteristisch für diese Phase ist die Schwäche, dass Gefahren bereits bekannt sein müssen,

[1] Social Engineering ist eine Methode mit der über 'Menschliche Komponenten' sicherheitsrelevante Daten in Erfahrung gebracht werden. Dies wird oft zur Vorbereitung von weiterführenden Angriffen verwendet. (https://www.security-insider.de/was-ist-social-engineering-a-633582/)

um verhindert werden zu können. Ein Prozess der Zugangskontrolle, sowohl durch bspw. Firewalls oder Mechanismen wie Active Directories[2] in der Windows Welt, kann nur die verbotenen Zugriffe behandeln, die als verboten bzw. nicht als erlaubt deklariert wurden. Verschafft sich ein Angreifer Zugang über Wege, die von der Zugangskontrolle als 'legal' bewertet werden, bspw. durch den Diebstahl eines Accounts mithilfe von Social Engineering, versagen reine Protect Mechanismen.
Auch im Falle der zuvor erwähnten Technologien wie Anti-Malware/Antivirus und IPS, werden nur solche Angriffsformen erkannt und letztlich verhindert, die durch Signaturen, Anomalie-Muster, Pattern oder Hash-Werte bereits bekannt und damit identifizierbar sind. Neue Angriffsformen, bspw. unbekannte Malware, die eine noch nicht bekannte Schwachstelle ausnutzt (Zero-Day-Exploit), wie es häufig bei APT-Angriffen der Fall ist, können nicht verhindert werden [Jangla u. a., 2015]. Das gilt insbesondere für die Vorbereitungsphasen von APT-Angriffen. Die im Smart Home klassischerweise vorhandenen Protect-Mechanismen werden in Kapitel 3 erläutert.

Detect

Die Detect Phase soll vordergründig Angriffe, welche die Protect Mechanismen überwunden haben, in einem möglichst frühen Stadium ihrer Durchführung erkennen. Dazu zählt, wie zuvor erwähnt, auch die Erkennung von Malware, Viren etc., aber auch die Identifikation von Angriffsmustern und Anomalien. Oftmals geht es dabei um das Erkennen von inneren Angriffen (Insider Threats). Ein Angriff von Innen ist klassischerweise ein Angriff, der von einer Person durchgeführt wird, welche legal Zugriff auf die IT-Infrastruktur hat (bspw. mit Hilfe von Social Engineering) [Blackwell, 2009]. Hat sich ein Angreifer, wie im Abschnitt der Protect Phase beschrieben, illegal Zugang verschafft, indem er Wege nutzt, die von den Protect Mechanismen als legal eingestuft werden, handelt es sich technisch ebenfalls um einen Insider Threat.
Die herkömmliche Erkennung solcher Angriffe geschieht auf der Basis von Intrusion Detection, sowohl durch Analyse des Netzwerkverkehrs als auch, wie bei host-basierten Intrusion Detection Systemen (IDS), durch Analyse von Prozessen auf einem Host [Jäger, 2015a]. Dabei ist ein IDS eine reine Detect Technologie, von der nur durch die Ableitung von React Maßnahmen ein Übergang in andere Phasen stattfindet. Die Abarbeitung der Detect Phase durch IDS hat jedoch zwei entscheidende Schwächen. Zum einen werden auch hier Angriffe durch bereits bekannte Muster und Anomalien identifiziert, sodass APT-Angriffe, sofern sie Zero-Day-Exploits

[2] Active Directories sind Verzeichnisdienste in denen Anmeldeinformationen gespeichert sind und verwaltet werden (https://technet.microsoft.com/de-de/library/dn151166.aspx)

ausnutzen, unerkannt bleiben. Zum anderen hat ein IDS lediglich einen eingeschränkten Blick auf einen speziellen Bereich der IT-Infrastruktur. So bleiben Angriffe unerkannt, die nur in Korrelation der Ereignisse an unterschiedlichen Stellen der Infrastruktur auffällig werden. Lösungen für dieses Problem könnten Ansätze wie beispielsweise Collaborative IDS sein, welche ihre Informationen an zentraler Stelle sammeln [Tan u. a., 2014]. Da jedoch auch bei Ansätzen wie diesem lediglich nach bereits bekannten Mustern im grundsätzlich gleichen Datentyp (hier Netzwerkverkehr) gesucht wird, ist eine Lösung notwendig, die sämtliche Informationen einer Infrastruktur berücksichtigt, um sie miteinander in Relation zu bringen und Erkenntnisse über möglicherweise erfolgte Angriffe abzuleiten.
An dieser Stelle kommen SIEM-Systeme zum Einsatz, deren Aufgabe es ist, Log-Daten aller möglichen Geräte und Anwendungen sowie den Netzwerkverkehr einer Infrastruktur zu sammeln und zu analysieren [Cappelli u. a., 2012]. Welche Geräte und Anwendungen im Smart Home notwendige Datenquellen sind, wird innerhalb dieser Arbeit auf Basis einer Auswahl an Angriffsszenarien definiert, wobei das Ziel ist, alle Datenquellen einzubinden, die nötig sind, um diese Angriffe zu erkennen (siehe Kapitel 6). Die genaue Funktionsweise von SIEM-Systemen wird im Abschnitt 2.2 behandelt.

React

In der React Phase werden Maßnahmen auf Basis von Ergebnissen aus der Detect Phase abgeleitet. Auch hier sind wieder die einfachen, automatischen React Prozesse der Anti-Malware/Antivirus Technologien zu erwähnen, welche eine Malware nach der Erkennung eines bekannten Musters entfernen.
Hauptsächlich geht es in der React Phase jedoch um manuelle Tätigkeiten, mit dem Ziel, einen erkannten Angriff möglichst schnell zu behandeln. Grundsätzlich gibt es zur Behandlung eines Angriffs oder Vorfalls zwei unterschiedliche Strategien. Zum einen kann versucht werden, den Angriff so schnell wie möglich zu unterbinden und damit die Infrastruktur vor weiterem Schaden zu bewahren. Das kann bspw. durch das Trennen von Netzwerkverbindungen, das Sperren von betroffenen Accounts, etc. geschehen. Zum anderen gibt es den strategischen Ansatz, den Angreifer zunächst nichts von seinem Auffliegen merken zu lassen, um weiter Spuren und Indizien sammeln zu können. Damit wird die Chance einer Identifikation und Rückverfolgung des Angreifers erhöht. Dahinter kann auch das Ziel stehen, genauer untersuchen zu können, ob bspw. bereits Daten abhanden gekommen sind. In beiden Fällen ist es wichtig, bereits im Voraus Reaktionen auf mögliche Angriffe oder Vorfälle zu definieren, um im Eintrittsfall schnell handlungsfähig zu sein und strukturiert und koordiniert vorgehen zu können [LaPiedra, 2002]. Diese vorbereitenden Maßnahmen zu treffen ist insbesondere

bei Zero-Day-Exploits verwendenden APT-Angriffen, also mehrstufigen Angriffen, die nicht vorausgesehen wurden, herausfordernd.
Für Aktionen der React Phase kann ein SIEM-System ein Lieferant an Ereignissen und Erkenntnissen sein, auf die es zu reagieren gilt. Bei der Übertragung des SIEM Ansatzes auf das Smart Home konzentriert sich diese Arbeit nicht auf die React Phase, will aber als Basis notwendige Informationen bereitstellen, um React Maßnahmen definieren und durchführen zu können. Die Definition von React Prozessen in diesem Kontext könnte auch Inhalt einer fortführenden Arbeit sein und wird in Kapitel 11 erwähnt.

Predict

In der Predict Phase werden Schlüsse aus den Erfahrungen der erfolgten Angriffe und Gegenmaßnahmen gezogen. Ziel ist es, auf Basis dieser Erkenntnisse Anpassungen und Verbesserungen der anderen Phasen vorzunehmen. Das kann bspw. eine Anpassung der Protect Maßnahmen, ein Umschwenken der im Fokus stehenden Informationsquellen und deren Bewertung in der Detect Phase oder eine verbesserte Vordefinition möglicher Reaktionen auf Angriffe oder Vorfälle in der React Phase sein [Davidson, 2017]. Innerhalb dieser Arbeit wird diese Phase, wie die React Phase, nicht vordergründig behandelt. Im Smart Home SIEM-Kontext könnte diese Phase bspw. das Verbessern oder Entwickeln zusätzlicher Use Cases[3] durch ausgiebige Testphasen beinhalten (siehe Kapitel 11).

2.1.1 Fokussierung auf Detect Phase

Bereits in der Einleitung wurde gezeigt, dass im Unternehmensumfeld derzeit eine verstärkte Aufmerksamkeit auf die Detect Phase gelegt wird und somit ein Umdenken stattfindet [IDC, 2017]. Im Folgenden werden nach einer Beschreibung des aktuellen Status bei der Umsetzung des Protect-Detect-React Ansatzes die Gründe für den steigenden Fokus auf die Detect Phase anhand von aktuellen Studien erörtert. Damit soll auch die wachsende Relevanz und Wichtigkeit des Einsatzes eines SIEM-Systems gezeigt werden.
Der aktuelle Status bei der Umsetzung des Protect-Detect-React Ansatzes lässt sich anhand einer Konferenzumfrage von ISACA und RSA aus dem Jahr 2016 festhalten. Dabei wurden hauptsächlich Vertreter von nordamerikanischen und europäischen Unternehmen, welche in einer breiten Branchenpalette angesiedelt sind, nach ihren Einschätzungen und Gefühlen bei der Umsetzung der Cybersecurity innerhalb ihres Unternehmens befragt [ISACA, 2016]. Grundsätzlich sind die Chefetagen der befragten Unternehmen sehr besorgt, was die Verletz-

[3] Erläuterung in Kapitel 2.2.1

barkeit des eigenen Unternehmens durch Cyberattacken angeht (36 % sehr besorgt und 46 % besorgt). Auch die Wahrscheinlichkeit, im Folgejahr angegriffen zu werden, wurde mit 74 % mit sehr wahrscheinlich oder wahrscheinlich beantwortet. Die kategorische Einteilung der in der Vergangenheit bereits erfolgten (und erfolgreichen) Angriffe ist ebenfalls zu betrachten. So sind 52 % der Angriffe im Bereich Malware angesiedelt, was bedeutet, dass Protect Mechanismen diese nicht aufhalten konnten. 41% wurden Opfer von Social Engineering Angriffen, welche zu Insider Threats führen können. Folglich werden verbesserte Detect Mechanismen notwendig. Mit den Fähigkeiten des Unternehmens bei der Umsetzung von Detect und React Maßnahmen fühlen sich zwar 31 % der Befragten gut, aber 42 % lediglich für simple Fälle. 22 % beantworteten diese Frage mit einem Nein oder zeigten sich unwissend.
Die Gründe für eine notwendige Fokussierung der Detect Phase und für die zuvor gezeigte Unbehaglichkeit in Security-Fragen gehen miteinander einher. Zum einen ist es die wachsende Anzahl, Diversität und Professionalität der Angriffe. Eine proofpoint Studie zeigt bspw. einen enormen Anstieg an Ransomware-Varianten seit 2015 [proofpoint, 2017]. Durch diese steigende Diversität und Professionalität muss man erkennen, dass Protect Barrieren grundsätzlich überwunden werden können und keinen verlässlichen Schutz mehr darstellen. Daraus leitet sich die Notwendigkeit neuer Detect Mechanismen ab [BSI, 2016]. Hinzu kommt, und das ist der zweite Grund, dass nach einer Studie des Ponemon Instituts IT-Angriffe im Finanzsektor erst nach durchschnittlich 98 Tagen (und bis zu 197 Tagen) entdeckt werden [Osborne, 2015].

SIEM-Systeme werden mittlerweile laut der zuvor erwähnten IDC Studie [IDC, 2017] bereits in 53 % der befragten Unternehmen als Detect Mechanismus eingesetzt. Ein Indikator, weswegen der SIEM Ansatz im Speziellen und Detect Mechanismen im Allgemeinen auch für das IoT und damit auch für den Bereich Smart Home notwendig werden könnten, zeigt sich in einem Umfrageergebnis von ISACA und RSA. Danach sind 53 % der befragten Unternehmen besorgt oder sehr besorgt, dass das IoT die Angriffsfläche der Unternehmen vergrößern wird. Lediglich 13 % zeigten sich im Angesicht dieser Entwicklung unbesorgt [ISACA, 2016]. In Kombination mit der in Kapitel 1 gezeigten wachsenden Relevanz des IoT und des Smart Home wird die Notwendigkeit entsprechender Maßnahmen gefestigt.

2.2 SIEM-Systeme

Nachdem Security Information and Event Management Systeme (SIEM) in den vorherigen Abschnitten in den Protect-Detect-React Ansatz eingeordnet wurden, wird im Folgenden auf die grundlegende Idee und Funktionsweise, inklusive einer Betrachtung der für den operativen

Betrieb relevanten Prozesse und Schwierigkeiten, eines SIEM-Systems eingegangen. Dabei stehen herkömmliche SIEM-Systeme im Fokus, wie sie bereits in Unternehmen oder für kritische Infrastrukturen betrieben werden. Ein Ziel dieser Arbeit ist es, die hier geschilderte Funktionsweise auf ein Smart Home SIEM-System zu übertragen.
Da SIEM-Systeme unterschiedlicher Anbieter architektonisch und technisch mitunter große Differenzen aufweisen, wird in diesem Kapitel nicht näher auf den technischen Aufbau eines SIEM-Systems eingegangen. Eine mögliche technische Variante eines SIEM-Systems wird bei der späteren Umsetzung des Smart Home SIEM-Systems in Kapitel 7 detailliert erläutert.
Grundsätzlich ist ein SIEM-System als Frühwarnsystem zu verstehen, dessen Ziel es ist, Erkenntnisse über bevorstehende oder bereits erfolgte Angriffe bereitzustellen. Die Art der erkannten Angriffe reicht dabei von einfachen, wie bspw. Brute Force Angriffen auf Benutzeraccounts [BSI, 2013], bis hin zu komplexen APT-Angriffen [Puri und Dukatz, 2015]. Die Erkenntnisse über Angriffe sollen aus dem Sammeln und Korrelieren unterschiedlichster Log-Daten einer IT-Infrastruktur gezogen werden. Unterschiedliche Methoden der Korrelation werden in Kapitel 6 genauer betrachtet. In der BSI Studie 'Studie über die Nutzung von Log- und Monitoringdaten im Rahmen der IT-Frühwarnung und für einen sicheren IT-Betrieb' [BSI, 2013] wird eine Vielzahl der üblicherweise verwendeten Typen von Log-Quellen aufgeführt. Wie im weiteren Verlauf der Arbeit deutlich wird, sind viele benötigte Log-Quellen einer Smart Home Umgebung dabei nicht berücksichtigt. Auf Abbildung 2.2 sind die Log-Quellen und Formate einer beispielhaften, typischen Unternehmensumgebung abgebildet. Das SIEM-System sammelt anhand unterschiedlicher Methoden (bspw. durch den Einsatz von Agenten, Netzwerksniffern oder APIs) unterschiedlichste Typen von Log-Daten. So sind hier bspw. aus dem Netzwerkbereich Log-Daten der klassischen Security-Systeme wie Firewalls, IDS/IPS und Proxy-Server aber auch Flow-Daten zur Analyse des Netzwerkverkehrs zu finden. Auf der Server Seite werden Ereignis-Logs, bspw. aus der Windows Welt gesammelt, um Anmeldeversuche zu erfassen, aber auch Logs von Antiviren-Scannern oder Mail-Servern. Eine zusätzliche Quelle können die regelmäßig produzierten Ergebnisse eines Schwachstellenscanners sein. Externe Threat Intelligence Feeds [4] geben bspw. aktuelle Blacklists von IP-Adressen, welche zu bekannten Botnetz-Servern gehören, in das SIEM-System, um so anhand von Korrelation verbotene Verbindungen aus der eigenen Infrastruktur erkennen zu können. Aus den Erkenntnissen der Korrelation des SIEM-Systems sollen in der React Phase Maßnahmen abgeleitet werden können, die mit einer möglichst geringen Reaktionszeit umgesetzt werden, um Schaden zu vermeiden oder zu begrenzen.

[4] Threat Intelligence Feeds sind automatisierte Benachrichtigungen von einer Cyber Threat Intelligence Plattform, welche dem Austausch von Gefährdungslagen und neuen Angriffsformen dient [Reiber, 2017].

Historisch entstanden SIEM-Systeme aus der Kombination zweier herkömmlicher Informationssysteme, dem Security Information Management (SIM) und dem Security Event Management System (SEM). SIM-Systeme haben eine zentralisierte Speicherung und Aufbewahrung von Log-Daten zum Ziel, wohingegen ein SEM-System eine Echtzeitwerkzeug ist, mit welchem Log-Daten im Zuge eines Monitoring analysiert werden können. Da zu den Zielen eines SIEM-Systems sowohl die echtzeitnahe Erkennung von Angriffen als auch die Archivierung von Log-Daten gehört, müssen SEM und SIM vereint werden. Zudem ist es notwendig, auch die Möglichkeit zu haben, Daten aus der Vergangenheit für SIEM-Korrelationen zu verwenden. Insbesondere APT-Angriffe erstrecken sich inklusive Vorbereitungsphase oftmals über einen längeren Zeitraum und können somit nicht echtzeitnahe erkannt werden. Zusätzlich können mithilfe eines SIEM-Systems auch forensische Aufgaben wahrgenommen werden, welche ebenfalls einen Zugriff auf länger gelagerte Daten erfordern [BSI, 2013]. Beim Testen des Übertragungsergebnisses innerhalb dieser Arbeit liegt der Fokus auf der echtzeitnahen Erkennung von Angriffen, wobei aber auch Daten aus der jüngeren Vergangenheit in die Korrelationen einbezogen werden.

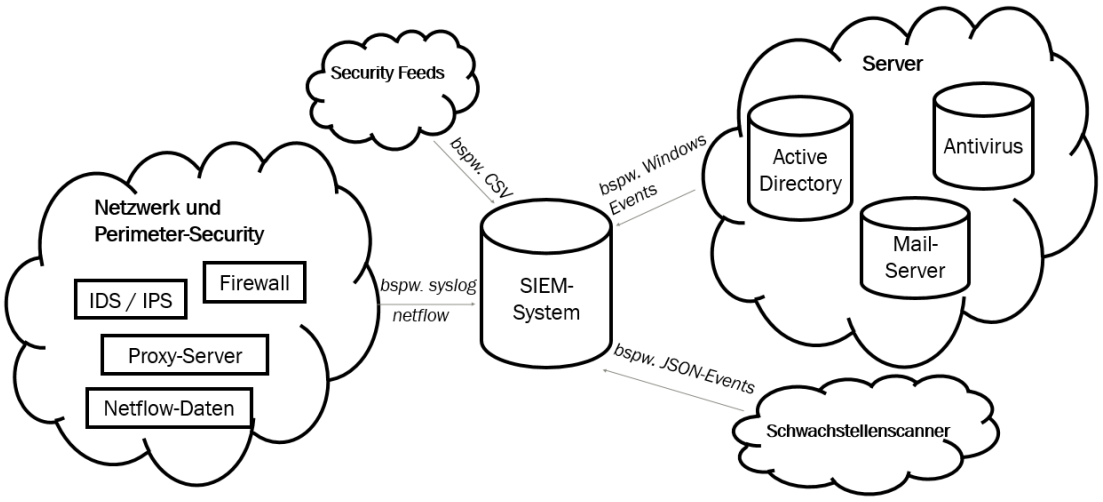


Abbildung 2.2: Typische Log-Quellen eines herkömmlichen SIEM-Systems (eigene Darstellung)

2.2.1 Funktionsweise

Um die Funktionsweise eines SIEM-Systems zu verstehen und später auf das Smart Home übertragen zu können, werden im Folgenden dessen unterschiedliche Arbeitsprozesse grob erläutert. In Bezug darauf findet die Strukturierung der Umsetzung des Smart Home SIEM-

Systems in Kapitel 7 statt. Dabei werden die Wichtigsten sinngemäß auf Basis der bereits erwähnten BSI Studie [BSI, 2013] unterteilt:

Sammeln

Im ersten Schritt werden Log-Daten von den ausgewählten, relevanten Quellen gesammelt. Dazu können bspw. Agenten verwendet werden, welche auf Hosts eingerichtet, die dort benötigten Log-Daten sammeln und zum SIEM-System weiterleiten. Beispiel hierfür ist das Sammeln von Windows Events von Windows-Servern oder das Sammeln von Log-Daten aus Verzeichnissen anderer Server. Auch Netzwerkverkehr kann mithilfe von Netzwerkagenten oder Agenten auf Routern, gesammelt werden. Eine weitere Methode ist das direkte Weiterleiten von Log-Daten an das SIEM-System. Voraussetzung dafür ist, dass der Host, dessen Log-Daten gesammelt werden sollen, in der Lage ist, diese weiterzuleiten. Das ist häufig bei zentralisierten syslog-Servern oder anderen Systemen wie Firewalls der Fall. Ein weiteres Beispiel ist das Ansprechen von APIs, um Log-Daten von Systemen abzurufen und in das SIEM-System zu importieren. Dieses Verfahren wird häufig beim Sammeln von Anwendungs-Logs verwendet, bspw. um die Ergebnisse eines Schwachstellenscanners abzurufen. Eine gute Übersicht über mögliche Wege der Anbindungen verschiedener Quellen und Datentypen bietet ebenfalls die erwähnte BSI Studie.

Filtern

Zu entscheiden, welche Logs einer bestimmten Quelle für ein SIEM-System relevant sind und welche überflüssig, ist ein wichtiger Prozess und eine wichtige Vorarbeit. Zum einen kann so ein zu hohes Volumen an Logs verhindert werden, welches Performance Probleme verursachen könnte [Butler, 2009]. Performance Probleme könnten wiederum zu einer Verzögerung der Angriffserkennung führen. Zum anderen erleichtert eine geringere, zielgerichtete Auswahl der Log-Daten die Arbeit auf den späteren Stufen der Normalisierung und Korrelation. Das Filtern von Events kann grundsätzlich an verschiedenen Stellen vorgenommen werden, abhängig von Log-Quelle und SIEM-Produkt. So sind in manchen Fällen Agenten bereits in der Lage, bestimmte Log-Daten zu verwerfen, bevor sie überhaupt in das SIEM-System gelangen. In anderen Fällen müssen die Logs erst am SIEM-System selbst verworfen oder gelöscht werden. Eine weitere Möglichkeit ist es, die Logs bereits beim Produzenten, also dem Host, dessen Logs abgegriffen werden sollen, zu filtern, indem sie gar nicht erst produziert werden. Dazu können bspw. Log-Level von Anwendungen heruntergestuft werden.

Normalisieren

Bei der Normalisierung werden zuvor gesammelte Log-Daten unterschiedlicher Log-Typen in ein einheitliches Format gebracht. Das ist Voraussetzung für eine spätere, erfolgreiche Korrelation über unterschiedliche Log-Quellen. Auf Abbildung 2.3 ist ein fiktiver, beispielhafter Log-Eintrag zu sehen. Dabei wird er im oberen Bereich im ursprünglichen und im unteren im normalisierten Zustand gezeigt. Beim Normalisierungsprozess werden zum einen die Namen der Log-Felder vereinheitlicht (bspw.: 'Time_and_Date' zu 'TIME_DATE'). Zum anderen werden Formatierungen der Daten selbst angepasst, bspw. die Formatierung des Datums.

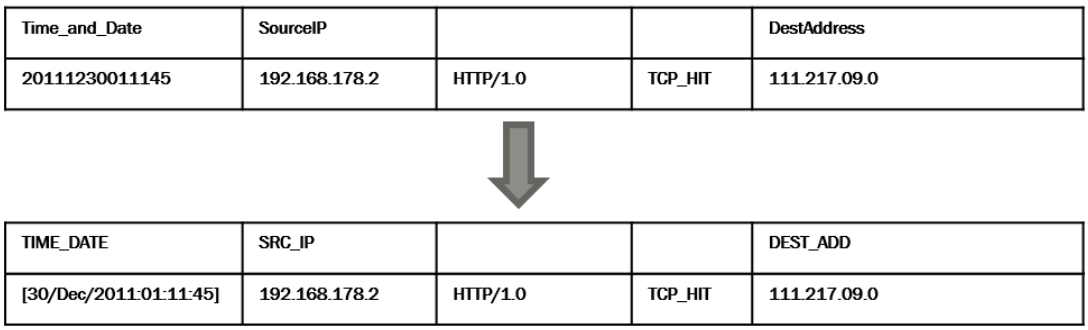


Abbildung 2.3: Normalisierung eines fiktiven Log-Eintrags (eigene Darstellung)

Korrelieren

Die Korrelation ist der entscheidende Schritt, wenn es um die automatisierte Erkennung von Angriffen durch ein SIEM-System geht. Ziel ist es, gesammelte Log-Daten nach Angriffsmustern oder Anomalien zu durchsuchen. Dabei soll es möglich sein, Log-Daten unterschiedlichster Typen miteinander zu korrelieren. Wird eine Auffälligkeit gefunden, erstellt das SIEM-System eine Benachrichtigung (im späteren Verlauf der Arbeit 'Notable Event' genannt) für den Betreiber des Systems, um über die mögliche Gefahrenlage zu informieren.
Korrelationsverfahren können unterschiedlicher Natur sein. So gibt es die Möglichkeit der regelbasierten Korrelation, welche Log-Ereignisse nach fest definierten Regeln durchsucht und verbindet. Statistische Korrelationen verwenden meist Schwellenwerte von Ereignissen, wobei bei einer Überschreitung ein Notable Event generiert wird. Eine dynamische Methode der Korrelation ist eine mit machine learning Methoden umgesetzte, statistische Variante. Hierbei lernt ein SIEM-System in einer Vorbereitungsphase das übliche Aufkommen bestimmter Log-Ereignisse. Wird dieses unerwartet über- oder unterschritten, wird ein Notable Event generiert

[Lavrova und Pechenkin, 2015]. Eine oder mehrere Korrelationen in Kombination werden in der Regel als Use Cases bezeichnet. Ein Use Case stellt dabei eine Angriffsstufe dar, die durch das SIEM-System aufgedeckt werden soll [Frye, 2009].
In der späteren Umsetzung des SIEM-Systems für Smart Homes werden unterschiedliche Korrelationsverfahren verwendet, um in Kapitel 8 passende Use Cases zur Erkennung der in Kapitel 6 ausgewählten Angriffsszenarien entwickeln zu können.

Priorisieren

Eine weitere wichtige Aufgabe eines SIEM-Systems liegt im Prozess der Priorisierung von generierten Notable Events. Da, je nach Größe der Infrastruktur, mitunter eine Vielzahl von Notable Events erzeugt werden könnte, ist es wichtig, diejenigen vorrangig zu betrachten, welche eine besonders hohe Kritikalität haben, um zeitnah passende React Maßnahmen ableiten und durchführen zu können. Dabei kann zum einen die Kritikalität des gefundenen Ereignisses selbst (Wie kritisch ist der entsprechende Use Case anzusehen? Wie weit ist ein Schwellenwert überschritten?) dienen. Zum anderen sollte ein SIEM-System eine Datenbank führen, in der alle Geräte der überwachten Infrastruktur aufgeführt sind - inklusive Prioritätsgrad des Systems. Somit würde ein überschrittener Schwellenwert beim Zugriff auf einen Domain Controller des Active Directory (siehe Abbildung 2.2) höher priorisiert werden als der selbe überschrittene Schwellenwert auf dem Computer eines einfachen Benutzers. Die Priorisierung wird bei der Umsetzung dieser Arbeit nicht näher behandelt, da der Fokus auf der Machbarkeit der Übertragung des Ansatzes liegt, aber in Kapitel 11 als zukünftige Notwendigkeit deklariert.

2.2.2 Operativer Betrieb

Der erfolgreiche operative Betrieb eines SIEM-Systems entscheidet sich vor allem an der Qualität und Ausführung definierter Prozesse innerhalb eines Unternehmens sowie der personellen Besetzung und Expertise. Diese Arbeit fokussiert zwar den technischen Ansatz und dessen Funktionalität, wird im Folgenden jedoch einen grundlegenden Überblick über den operativen Betrieb eines SIEM-Systems skizzieren. Aufgenommen werden die betrieblichen Aspekte vor allem im Ausblick dieser Arbeit (Kapitel 11), um den Bedarf zukünftiger Betrachtungen dieses Aspekts deutlich zu machen.
Grundsätzlich kann der operative Betrieb eines SIEM-Systems in zwei Kategorien aufgeteilt werden. So fallen administrative Aufgaben an, welche sich in der Wartung, Instandhaltung und Pflege des SIEM-Systems selbst zeigen [Mahrenholz und Schuhmann, 2017].
Der wesentliche Teil des operativen SIEM-Betriebes ist jedoch im Protect-Detect-React Ansatz verwurzelt. Von einem SIEM-System produzierte Notable Events (Detect), müssen zu einem

React Vorgang führen, genauer einem Incident Response Prozess. Der Incident Response Prozess wird von einem Security Operations Center (SOC) betrieben. Ein SOC ist im Idealfall eine 24/7 besetzte Abteilung, welche Notable Events, die unter anderem aus einem laufenden SIEM-System zum Vorschein kommen, entgegennimmt und weiterverarbeitet. Diese Verarbeitung erfolgt dabei in der Regel mehrstufig. Ziel ist es, Incidents abzuarbeiten, indem sie auf den ersten Bearbeitungsstufen zusammengefasst und mit weiteren Kontext-Informationen angereichert, priorisiert und auf False Positives gefiltert werden. Anschließend kann ein Incident von Security Analysten genauer untersucht werden, um gegebenenfalls React Maßnahmen einzuleiten. Um ein SOC effektiv betreiben zu können, muss ein hohes Fachwissen vorhanden sein, sowohl was grundsätzliche Security Themen und bspw. Netzwerkanalyse betrifft, als auch die Kenntnis des betriebenen SIEM-Systems und der überwachten Infrastruktur [Torres, 2015].

Neben der Bewältigung des Incident Response Prozesses, führen auch die im vorherigen Abschnitt erläuterten funktionalen Prozesse des SIEM-Systems zu manuellen Aufgaben. Die angeschlossenen Datenquellen müssen ausgewählt und stetig kontrolliert werden. Entstehen Änderungen in der IT-Infrastruktur, müssen neue Systeme als potentielle Log-Quellen in Betracht gezogen und bewertet werden. Das Filtern und Normalisieren der Daten muss bei Änderungen an Quell-Systemen, bspw. aufgrund von Updates durch die das Log-Format verändert wird, ebenfalls angepasst werden. Die Hauptaufgabe in diesem Bereich stellt allerdings die Pflege und Entwicklung der Use Cases dar. Dazu muss sich das SOC stetig mit aktuellen Gefahren auseinandersetzen, diese im Kontext der eigenen Infrastruktur bewerten und gegebenenfalls in Korrelationsregeln umsetzen [Srinivas, 2014]. Die in diesem Abschnitt genannten Aufgaben müssen auch im Rahmen dieser Arbeit beim Aufsetzen des SIEM-Systems sowie der Entwicklung der Use Cases initial durchgeführt werden. Der hier beschriebene kontinuierlich notwendige Prozess wird nicht näher betrachtet.
Mit der Betrauung dieser Aufgaben ist ein SOC ein wesentlicher Bestandteil des Protect-Detect-React Ansatzes und essentiell für den operativen Betrieb eines SIEM-Systems, welches ohne SOC als Detect Technologie keinen Mehrwert für die Sicherheit einer Infrastruktur entwickeln kann.

2.2.3 Schwierigkeiten

Die hauptsächlichen Schwierigkeiten zeigen sich vor allem durch hohen manuellen Aufwand bei der Einrichtung und dem operativen Betrieb eines SIEM-Systems. In den Kapiteln 10 und 11 wird auch Bezug auf den notwendigen Aufwand zum Aufsetzen und Betreiben eines Smart Home SIEM-Systems genommen. Dieser Aufwand bezieht sich hauptsächlich auf die zuvor

beschriebenen Tätigkeiten eines SOC und wird in diesem Abschnitt nicht weiter behandelt. Jedoch entstehen zusätzliche Herausforderungen beim Einsatz eines herkömmlichen SIEM-Systems. Ein sensibles Thema ist der Datenschutz. Log-Daten unterschiedlichster Systeme enthalten mitunter sensible Informationen, die auf Personen zurückzuführen sind. Hier gilt es zum einen die Integrität der Daten während des Transports zum SIEM-System zu wahren. Zum anderen ist es wichtig, die Speicherung der Daten im SIEM-System selbst entsprechend sensibel zu behandeln. Im Unternehmenskontext muss meist ein Betriebsrat, sofern vorhanden, im Bilde über die Verwendung dieser Daten sein. Oft werden zur Behandlung dieser Problematik Anonymisierungs- oder Pseudonymisierungsmethoden eingesetzt, welche jedoch nicht die Auswertung der Daten für einen SOC-Mitarbeiter unmöglich machen dürfen [BSI, 2013].
Ein weiteres Problem ist die Verfügbarkeit, Vollständigkeit und Integrität der gesammelten Daten. So kann ein SIEM-System nur dann zuverlässig arbeiten, wenn die gelieferten Daten diese Eigenschaften erfüllen. Insbesondere die Verfügbarkeit und die Vollständigkeit entwickeln sich bei Anwendungs-Logs zu einem Problem. So ist ungenügendes Logging bereits auf Platz 10 der 'OWASP Top 10 2017: The Ten Most Critical Web Application Security Risks' [van der Stock u. a., 2017].
Z. Tan et al haben in ihrer Arbeit 'Enhancing Big Data Security with Collaborative Intrusion Detection' [Tan u. a., 2014] einen zentralisierten IDS Ansatz geschaffen und gezeigt, dass ein Angriff auf ein solches System von besonderer Kritikalität ist, wobei hier lediglich Daten unterschiedlicher Intrusion Detection Systeme gesammelt und korreliert werden. Ein SIEM-Sytem verschärft diese Situation aufgrund der vielen zusätzlichen Datenquellen noch weiter. Dabei ist zum einen die Verfügbarkeit von entscheidender Rolle. Fällt ein SIEM-System im Angriffsfall aus, könnte das zur Blindheit des SOC führen. Ein notwendiger, schneller React Prozess auf Folgeangriffe könnte nicht abgeleitet und ausgeführt werden. Zum anderen ist die Kompromittierung des SIEM-Systems selbst ein mögliches Ziel von Angreifern. Auf einem SIEM-System lagern große Mengen an sensiblen Daten, die abgesehen von den bereits erwähnten datenschutzrelevanten Gesichtspunkten entscheidende Informationen über die IT-Infrastruktur in sich bergen. Beispielhaft zu nennen sind die Schwachstellen aller Systeme durch die importierten Ergebnisse eines Schwachstellenscanners.
Zusätzlich zu den für einen operativen SIEM-Betrieb notwendigen Prozessen und Sicherheitsrisiken, gibt es auch technische Herausforderungen zu meistern. Im Vordergrund steht neben der Betreuung der Systeme selbst vor allem das hohe Log-Volumen eines Unternehmens. Daraus folgt zum einen ein hoher Bedarf an Speicherplatz. Zum anderen kann eine hohe Belastung der Bandbreite des Unternehmensnetzes entstehen. So muss ein Unternehmen mit einem Log-Aufkommen von ungefähr 20 000 Einträgen pro Sekunde bereits mit 172,8 GB benötigtem

Speicherplatz pro Tag rechnen [Butler, 2009]. Diese Herausforderungen verdeutlicht die notwendige sorgfältige Arbeit im Filter Prozess eines SIEM-Systems.
Ein Abgleich der in diesem Abschnitt erwähnten Schwierigkeiten beim Einsatz herkömmlicher SIEM-Systeme mit einem möglichen Smart Home SIEM-System wird in Kapitel 3 vorgenommen.

2.3 Bekannte Hersteller

Zum Abschluss dieses Kapitels wird ein kurzer Überblick bekannter Hersteller aktueller, herkömmlicher SIEM-Systeme gegeben, welche von Gartner, wie auf Abbildung 2.4 zu sehen, im September 2017 eingeordnet wurden. Der abgebildete Gartner Quadrant ist dabei als Indikator für aktuelle Trends auf dem SIEM-System Markt zu sehen.
Besonders hervorzuheben sind dabei die Lösungen im rechten oberen Abschnitt. Dabei ist Splunk[5] als allgemein verwendbares Big Data Tool zu sehen, welches durch eine zusätzliche Komponente mit SIEM-Funktionalitäten ausgestattet wurde. Die Lösungen QRadar[6] (IBM) und LogRhythm[7] sind dagegen gezielt entwickelte SIEM-Systeme.

[5] https://www.splunk.com/de_de
[6] https://www.ibm.com/de-de/marketplace/ibm-qradar-siem
[7] https://logrhythm.com/

Source: Gartner (December 2017)

Abbildung 2.4: Gartner 2017 Magic Quadrant for SIEM [Gartner]

3 Smart Home

Das Smart Home kann wegen seiner zugrunde liegenden Technologien und Funktionsweisen als ein Anwendungsgebiet des Internet of Things (IoT) betrachtet werden. Um ein Verständnis der Gefährdungslage von Smart Homes zu gewinnen und für dieses Gebiet einen SIEM Ansatz zu entwickeln ist es somit notwendig, sich zunächst mit der Architektur, Technologie und Funktionsweise des Internet of Things auseinanderzusetzen.
Um dieser Herangehensweise zu folgen, vermittelt dieses Kapitel zunächst Grundlagen des Internet of Things, um daraufhin den Anwendungsfall Smart Home im Speziellen zu betrachten und einzuordnen. Im darauf folgenden Abschnitt wird auf die Sicherheit von Smart Homes eingegangen, indem gezielt das Smart Home betreffende Untersuchungen herangezogen und Aspekte der generellen IoT-Security eingebracht werden.
Auf Grundlage aus der bereits in Kapitel 2 gezeigten Funktionsweise eines SIEM-Systems werden in Kombination mit den in diesem Kapitel gewonnenen Erkenntnissen über die Beschaffenheit des IoT und des Smart Home Schlüsse auf ein mögliches Smart Home SIEM-System gezogen. Dazu gehört eine Betrachtung der bereits abzusehenden Herausforderungen, im Abgleich mit den Herausforderungen herkömmlicher SIEM-Systeme (siehe Kapitel 2), sowie die Erläuterung bereits vorhandener Ansätze anderer Autoren. Im darauf folgenden Kapitel 4 wird der eigene Ansatz als Kombination und Weiterentwicklung der bereits vorhandenen Ansätze und gezogenen Erkenntnisse skizziert.
Ein weiteres Ziel dieses Kapitel ist es, die in Kapitel 1 geforderte ganzheitliche Betrachtung des Smart Home durch ein Smart Home SIEM-System in einem ersten Schritt zu definieren. Dazu wird in Abschnitt 3.1.1 ein IoT Architekturmodell ausgewählt, welches sich im weiteren Verlauf der Arbeit auf das Smart Home übertragen lässt. Werden alle Ebenen des Architekturmodells von der entwickelten SIEM-Lösung abgedeckt, kann von einer ganzheitlichen Betrachtung gesprochen werden.

3.1 Internet of Things & Smart Home

Das Smart Home basiert auf dem Ansatz, der Architektur und den Technologien des IoT. Zunächst werden die IoT Grundlagen hauptsächlich auf Basis der Arbeiten 'The Internet of

Things: A survey' von Luigi Atzori et. al [Atzori u. a., 2010] und 'Architecting the Internet of Things: State of the Art' von Mohammed Riyadh Abdmeziem et. al [Romdhani u. a., 2015] erläutert, um diese im Anschluss auf das Smart Home projizieren zu können.

3.1.1 IoT Grundlagen

Auf der Suche nach einer eindeutigen Definition des Internet of Things stößt man auf unterschiedliche Ergebnisse. So wird das IoT bspw. als Interaktion zwischen der physischen und der digitalen Welt beschrieben, in der die physische Seite mit Hilfe von Sensoren und Aktoren (Sensoren und Aktoren werden im späteren Verlauf erläutert) mit der digitalen kommuniziert. Andere sprechen von einem Paradigma, in welchem alle möglichen Netzwerk- und Computer-Mechanismen in Objekte integriert werden [Sethi und Sarangi, 2017]. Die am besten für den Zusammenhang mit einem Smart Home geeignete Definition beschreibt das Internet of Things als eine allgegenwärtige Präsenz von Dingen in unserer physischen Umgebung, welche in der Lage sind miteinander zu kommunizieren, zu interagieren und zu kooperieren, um gemeinsame Ziele zu erreichen [Atzori u. a., 2010].
Mittlerweile gibt es eine Vielzahl an Anwendungsgebieten, die auf IoT-Technologien basieren. Aus der Perspektive von Unternehmen sind mit IoT-Komponenten, in diesem Fall RFID-Chips (auf RFID wird später genauer eingegangen), ausgestattete Güter, die in Liefer- und Versorgungsketten verfolgt werden können [Romdhani u. a., 2015] beispielhaft zu nennen aber auch der Einsatz in Industrieanlagen, zur Verbesserung der Produktivität durch Robotik [Atzori u. a., 2010]. Im privaten und sozialen Umfeld erobern IoT-Komponenten den gesundheitlichen Bereich, indem bspw. Vitalparameter von Patienten durch Sensoren gesammelt und automatisch an Ärzte übermittelt werden. Im Social Media Bereich erzeugen IoT-Komponenten bspw. Tweets über ihren eigenen Zustand. Ein weiterer dieser privaten Anwendungsbereiche ist der hier fokussierte Bereich des Smart Home, auf welchen später genauer eingegangen wird [Romdhani u. a., 2015].
Auch wenn meist von 'dem' Internet of Things gesprochen und geschrieben wird, ist das Internet of Things keineswegs als einheitliche Technologie oder System zu betrachten. Vielmehr beschreibt es eine Vielzahl an Technologien und Geräten, die miteinander agierend das Internet of Things bilden. Unterschieden wird hier zunächst zwischen Komponententypen mit jeweils zugewiesenen Rollen. Eine Komponente entspricht dabei nicht zwangsläufig genau einem Gerät. Ein physisches Gerät kann auch mehrere unterschiedliche Komponenten enthalten. Die Komponenten werden in Sensoren, Aktoren, Prozessoren und Sender oder Empfänger unterteilt. Sensoren und Aktoren sind diejenigen Geräte, die mit der physischen Umwelt interagieren. Sensoren sammeln dabei Informationen aus der Umwelt, wie Messdaten oder Zustände von

Geräten. Aktoren nehmen aktiven Einfluss auf die Umwelt, indem sie nach einem Kommando bspw. Geräte an- und ausschalten oder steuern. Wenn im Allgemeinen von IoT-Objekten gesprochen oder geschrieben wird, sind meist Geräte gemeint, die mit Sensoren oder Aktoren ausgestattet sind. Prozessoren dagegen sind Komponenten, die die von Sensoren empfangenen Daten oder Steuerbefehle verarbeiten und über die Sender und Empfänger weiterschicken [Sethi und Sarangi, 2017].

Der Versuch, dem Internet of Things ein einheitliches Architekturmodell zu verpassen, dauert noch immer an. Meist wird eine Architektur aus drei Layern vorgeschlagen [Romdhani u. a., 2015], die auch in dieser Arbeit aufgrund der besten Kompatibilität mit dem Smart Home gewählt wird. Auch wurden wichtige IoT-Security Betrachtungen auf Basis des 3-Layer-Modells vorgenommen, bspw. die Arbeit von Qi Ling et al. 'Security of the Internet of Things: perspectives and challenges' [Jing u. a., 2014], welche im späteren Verlauf des Kapitels verwendet wird. Einige andere Modelle sind bspw. aus sechs Layern aufgebaut [Sethi und Sarangi, 2017]. Grundsätzlich zeigt sich hier eines der Problemfelder des IoT. Durch die Diversität der Anwendungsgebiete und der damit einhergehenden Diversität der Funktionsweisen und Anwendungen fällt es schwer, alle Möglichkeiten unter einer Architektur zu vereinen. Im Rahmen dieser Arbeit wird die 3-Layer-Architektur näher betrachtet. Damit erfolgt ein wichtiger Schritt zur geforderten Definition nach einer ganzheitlichen Betrachtung des Smart Home mithilfe des SIEM Ansatzes. So ist die Voraussetzung, dass nach einer Übertragung des Architekturmodells auf die Smart Home Versuchsumgebung (siehe Kapitel 5), alle Layer von der SIEM-Lösung einbezogen werden.

Die 3-Layer-Architektur besteht aus Perception, Transport und Application Layer (siehe Abbildung 3.1). Im Laufe der Arbeit, insbesondere bei der konkreten Übertragung der 3-Layer-Architektur auf die Versuchsumgebung des Smart Home (Kapitel 5), wird sich herausstellen, dass die Grenzen der drei Layer leicht verschwimmen und Technologien wie Protokolle nicht zwangsläufig auf einen Layer begrenzt sind.

Der Perception Layer ist vordergründig mit der Aufgabe betraut, die Umwelt wahrzunehmen und Objekte zu identifizieren, die zum IoT gehören. Auf diesem Layer kommt eine besondere Eigenschaft von IoT-Objekten zum Tragen: IoT-Objekte laufen meist batteriebetrieben und müssen daher mit möglichst geringem Energieverbrauch auskommen. Aufgrund dieser Tatsache werden auf dem Perception Layer vordergründig Technologien wie Radio-Frequency IDentification (RFID) verwendet. RFID ist eine Technologie die es ermöglicht, Mikrochips, die gespeicherte Informationen in sich tragen, in physische Objekte zu verbauen. Diese Informationen können zur Identifikation des Objekts genutzt werden. Ein RFID-Chip kann sowohl Signale empfangen als auch die auf ihm gespeicherten Informationen zurücksenden . So wird

es IoT-Komponenten möglich, Objekte in ihrer Umwelt zu identifizieren. Verbreitete Alternativen sind das Wireless Sensor Network (WSN), Global Positioning System (GPS) und Near Field Communication (NFC). Auf die genaue Funktionsweise der einzelnen Technologien wird an dieser Stelle nicht eingegangen. Die in der Versuchsumgebung dieser Arbeit (Kapitel 5) betroffene Variante wird bei der Auswahl von Angriffsszenarien im Smart Home SIEM (siehe Kapitel 6) erläutert.
Eine weitere Aufgabe des Perception Layer besteht im Sammeln von Informationen aus der physischen Umwelt (mithilfe von Sensoren) und deren Digitalisierung [Romdhani u. a., 2015]. Dabei darf nicht vergessen werden, dass auch die entgegengesetzte Richtung, also das Übersetzen von Informationen bzw. Kommandos in Aktionen auf die Umwelt, also das Umwandeln von digitalen Informationen in physische Aktionen der Aktoren, in den Perception Layer gehört.

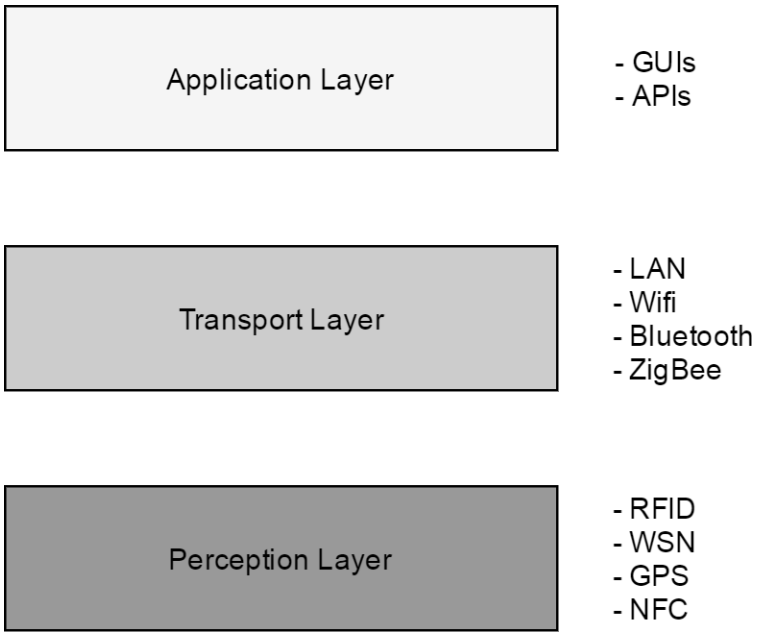


Abbildung 3.1: IoT 3-Layer-Architektur [nach [Romdhani u. a., 2015]]

Der Transport Layer dient der Verarbeitung der Daten aus dem Perception Layer und deren Übermittlung mithilfe verschiedener Netzwerktechnologien. Dazu gehören Local Area Networks (LAN) und kabellose Verbindungen. Auch hier kommt die im Zusammenhang mit dem Perception Layer bereits erwähnte Herausforderung des möglichst niedrigen Energieverbrauchs zum Tragen. So wurden neue Protokolle für kabellose Verbindungen entwickelt, die nicht mehr wie Wifi auf dem IEEE Standard 802.11 sondern auf IEEE 802.15.4 basieren und

energiesparend arbeiten [Sethi und Sarangi, 2017]. Auf eines dieser Protokolle, das ZigBee Protokoll, wird bei der Auswahl der Angriffsszenarien zur Use Case Entwicklung genauer eingegangen (siehe Kapitel 6). Ein weiteres populäres Beispiel ist die Verwendung von Bluetooth.

Der Application Layer dient als Schnittstelle zu den Benutzern des IoT. Hier werden verarbeitete Daten aus dem Transport Layer verwendet. Der Application Layer bietet Anwendungen bspw. in Form von Webanwendungen oder Mobile Apps an, über die der Benutzer von Sensoren gesammelte und weiterverarbeitete Daten einsehen oder IoT-Aktoren steuern kann [Romdhani u. a., 2015]. Eine wichtige Ergänzung für die spätere security-bezogene Betrachtung des Layers ist, dass dem Application Layer nicht nur übliche Front-End Anwendungen in Form von Graphical User Interfaces (GUI) zugesprochen werden, sondern auch Application Programming Interfaces (APIs) ohne implementiertes Front-End. Hierüber kann eine Anwendung bspw. mithilfe einer anderen Anwendung angesprochen, ausgelesen oder gesteuert werden.
Wie in diesem Abschnitt aufgezeigt wurde, werden im IoT auf sämtlichen Layern unterschiedliche Technologien verwendet. Eine herstellerübergreifende Standardisierung findet kaum statt. Das führt neben der bereits beschriebenen Problematik des einheitlichen Architekturmodells auch zu Security-Herausforderungen, auf die im weiteren Verlauf dieses Kapitels eingegangen wird.

3.1.2 Smart Home Grundlagen

Nachdem im vorherigen Abschnitt das Internet of Things anhand von Architektur und einem Überblick über die verwendeten Technologien dargestellt wurde, fokussiert dieser Abschnitt das Smart Home, als ein spezielles Anwendungsgebiet des IoT.
Das Smart Home wird von Mussab Alla et al. in 'A review of smart home applications based on Internet of Things' [Alaa u. a., 2017] als ein Netzwerk physischer Elektrogeräte, Sensoren, Software und Netzwerkverbindungen innerhalb eines physischen Zuhauses beschrieben. Wie in der Einleitung dieser Arbeit bereits erwähnt, gibt es unterschiedliche Ziele, die bei einer Smart Home Implementierung verfolgt werden können. So gibt es Ansätze, welche zum Ziel haben, ein altersgerechtes Wohnen für diese wachsende Bevölkerungsgruppe durch 'smarte' Komponenten zu ermöglichen. Dabei wird in viele Bereiche des Wohnens eingegriffen: Von der Sicherheit, bspw. durch Bewegungssensoren und dem Erkennen von unerwünschtem Personen im Haus bis hin zur Mundhygiene durch Überwachung der Zahnpflegegewohnheiten der Bewohner, wird vor kaum einem Gebiet Halt gemacht [Kon u. a., 2017]. Andere Ansätze sehen ein verbessertes Energiemanagement als Ziel eines Smart Home [Allerding und Schmeck, 2011].

Der häufigste Anwendungsfall ist allerdings durch den privaten Haushalt abgedeckt, der nach und nach mit IoT-Komponenten ausgestattet wird und sich somit, vielleicht ohne dass sich dessen Bewohner darüber bewusst sind, in ein Smart Home verwandelt. Das Anliegen dieses Smart Home Ansatzes ist hauptsächlich ein gesteigerter Komfort aber auch erhöhte Sicherheit [Alam u. a., 2012]. Zu den hier typischerweise eingesetzten Smart Home Geräten zählen bspw. Rauchmelder, Türschlösser, Bewegungssensoren, Heizungssteuerung, Lichtanlagen und Fernseher. Diese Geräte werden meist über Front-End Anwendungen vom Bewohner gesteuert und überwacht. Die vom Bewohner gesteuerte Anwendung sendet die Anforderung an ein sogenanntes Gateway, meist ein physisches Gerät, welches direkt per WLAN oder LAN mit dem Heimnetz verbunden ist. Gleichzeitig kommuniziert das Gateway mit den IoT-Objekten über IEEE 802.15.4 Verbindungen und steuert sie über diesen Weg [Alaa u. a., 2017]. Der Bewohner hat dazu meist eine browserbasierte Webanwendung oder eine Smart Phone App zur Verfügung. Dabei gibt es drei unterschiedliche Kommunikationsmodelle: Direct, Transit und External (siehe Abbildung 3.2). Beim direkten Kommunikationsmodell (a) steuert der Bewohner die Anwendung, hier durch ein Smartphone dargestellt, welche direkt auf das IoT-Gerät zugreift und dieses steuert. Das IoT-Gerät sendet nun bspw. Statusmeldungen an Server des jeweiligen Herstellers ins Internet. Beim zweiten Modell (b) ist das IoT-Gerät selbst nicht in der Lage einen Server im Internet zu kontaktieren, also steuert die Anwendung das Gerät und benachrichtigt den Herstellerserver direkt. Beim dritten Modell (c) steuert der Bewohner das Gerät über einen Server des Herstellers. Das ist meist dann der Fall, wenn der Bewohner seine Smart Home Geräte steuert, ohne sich selbst im Heimnetz zu befinden [Notra u. a., 2014]. Diese Arbeit wird sich bei der Betrachtung und Umsetzung eines SIEM-Systems in diesem Umfeld auf das direkte Kommunikationsmodell (a) fokussieren.

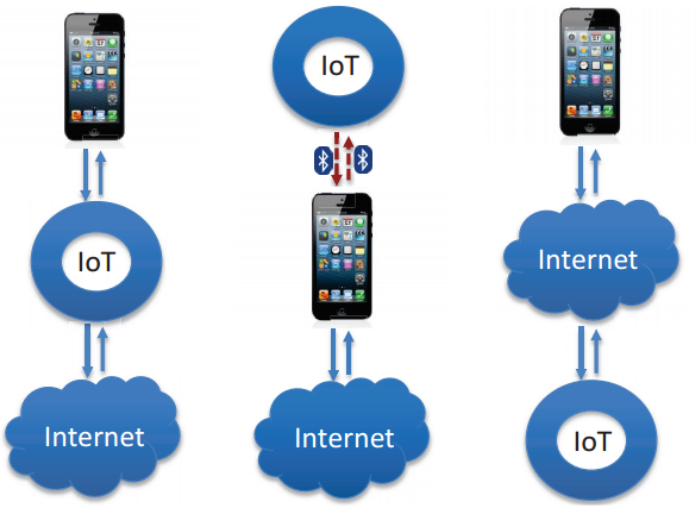


Abbildung 3.2: Kommunikationsmodelle im Smart Home [Notra u. a., 2014]

Auch das Anwendungsgebiet des Smart Home kämpft mit vielen allgemeinen Problemen des IoT. Da auch hier von Gerät zu Gerät unterschiedliche Protokolle und Technologien auf den drei IoT-Layern (Abbildung 3.1) verwendet werden, entwickelt jeder Hersteller seine eigene Steueranwendung, sein eigenes Gateway und verwendet letztlich auch sein eigenes Kommunikationsmodell (Abbildung 3.2). Auch hier ist also eine Vereinheitlichung oder Standardisierung noch nicht erfolgt [Jacobsson u. a., 2014].

3.2 Security im IoT & Smart Home

Nachdem in den beiden vorangegangenen Abschnitten ein Überblick über Ziele, Architektur und Technologie des IoT insgesamt und Smart Home im Speziellen gegeben wurde, wird in diesem Abschnitt auf die daraus entstehenden Risiken eingegangen. Viele Security-Probleme des IoT decken sich mit denen des Smart Home, andere wiederum sind konkret auf Basis von Smart Home Komponenten und deren Gegebenheiten ergründet worden.
Um sich den Security-Problemen des Smart Home zu nähern, muss zunächst ein Überblick über die möglichen Auswirkungen eines Angriffs geschaffen werden. Die Arbeit 'How to Increase the Security of Smart Buildings' von Steffen Wendzel [Wendzel, 2016] schlägt vor, die Auswirkungen von Angriffen in aktiv und passiv zu gliedern.
Aktive Auswirkungen beeinflussen den 'Betrieb' des Smart Home, indem der Angreifer Ak-

tionen der IoT-Aktoren herbeiführt. Dazu zählt die Manipulation von IoT-Objekten wie Beleuchtung [Sivaraman u. a., 2016], sodass bspw. Glühbirnen ein- und ausgeschaltet werden können. Wird dies mit einer bestimmten Blinkfrequenz ausgeführt, können in extremen Fällen epileptische Anfälle der Bewohner die Folge sein [Ronen u. a., 2017]. Auch andere, sensible Infrastrukturen wie Elektrizität, bspw. in Form von 'smarten' Steckdosen, und die Wasserversorgung sind gefährdet [Notra u. a., 2014]. Besonders kritisch werden aktive Auswirkungen, wenn es um das Manipulieren von Sicherheitsmechanismen wie 'smarten' Türschlössern geht, wodurch sich der Angreifer physischen Zugang zum Smart Home verschaffen kann [Rose und Ramsey, 2016]. Aktive Auswirkungen wie die hier genannten, können also ein durch Security Angriffe ausgelöstes Safety Risiko haben, welches physischen Schaden verursachen kann [Jäger, 2015b].

Passive Auswirkungen gehen meist mit Datenschutzthematiken einher. So werden durch das Kompromittieren von IP-Kameras oder Mikrofonen, bspw. Babyphones, ausführliche Spionageaktivitäten möglich. Durch das Hinzunehmen von Bewegungssensoren oder Messgeräten von Gesundheitsparametern sind sensible Informationen über die Bewohner des Hauses, wie ihre aktuelle Position oder ihr Gesundheitszustand, in Gefahr [Notra u. a., 2014]. Dabei ist anzumerken, dass im Gegensatz zum herkömmlichen Ausspionieren von Internetaktivitäten auch Offline-Aktivitäten durch die genannten IoT-Sensoren verfolgt werden können [Apthorpe u. a., 2017b].
Passive und aktive Auswirkungen können auch in Kombination auftreten. So könnte passives Auskundschaften der Position der Bewohner über Kameras oder Bewegungssensoren den perfekten Zeitpunkt bestimmen, um das 'smarte' Schloss des Smart Home zu manipulieren und damit einem Einbruch freien Raum zu geben. Auch können vom Angreifer erlangte Informationen über Smart Home Komponenten auf dem Schwarzmarkt verkauft werden. In bereits aufgetretenen Fällen wurden derartige Geräte zu einem Botnetz verbunden, um Dritte anzugreifen oder Spam-Mails zu versenden [Sivaraman u. a., 2016].
Im Rahmen dieser Arbeit werden für die Umsetzung des Smart Home SIEM-Systems exemplarisch Angriffsszenarien ausgewählt und betrachtet, welche Angriffsstufen mit sowohl aktiven als auch passiven Auswirkungen beinhalten (siehe Kapitel 6).
Nach der Betrachtung möglicher Risiken muss nach den Gründen gesucht werden, aufgrund derer das IoT im Allgemeinen und das Smart Home im Speziellen eine Angriffsfläche bietet und derartige Attacken möglich macht.
Eine der Hauptursachen für die hohe Risikosituation im IoT und Smart Home ist bei den Herstellern der jeweiligen Produkte zu finden. Hier gibt es verschiedene Aspekte zu betrachten. Das

bereits in vorherigen Abschnitten angesprochene Problem der Heterogenität und Diversität der IoT-Geräte führt neben Architektur- und Designproblemen auch zu Sicherheitsrisiken. So entsteht aufgrund der Tatsache, dass die meisten Hersteller ihre eigene komplette Lösung inklusive IoT-Objekt und Gateway verkaufen, ein breites Spektrum an stark individuellen Anwendungen, APIs, etc. Diese Beschaffenheit macht es bislang schwierig bis unmöglich, eine gesamtheitliche Risikoanalyse durchzuführen, aus welcher Protect-Maßnahmen abgeleitet werden könnten [Jacobsson u. a., 2014]. Des weiteren sind die meisten Hersteller nachlässig, wenn es um die Betrachtung und Implementierung von Sicherheitsmechanismen in ihren Produkten geht. Das hat meist wirtschaftliche Gründe wie bspw. hoher Druck, schnell Produkte auf den Markt zu werfen und keine Ressourcen für teure Security-Maßnahmen bereitzustellen. Auch verdienen Hersteller in den derzeitigen Geschäftsmodellen lediglich an der verkauften Stückzahl, nicht am kontinuierlichem Service, wie es bei typischen Produkten im Unternehmensumfeld der Fall ist [Sivaraman u. a., 2016]. Das führt zum einen dazu, dass insbesondere die Anwendungen auf den Gateways und die Anwendungen für den Endanwender (den Bewohner) nicht ausreichend geschützt werden, das Logging vernachlässigt wird oder gar nicht vorhanden ist. Die Smart Home Risikoanalyse 'On the Risk Exposure of Smart Home Automation Systems' von Andreas Jacobsson et al. [Jacobsson u. a., 2014] hat nach 'Human Related' Risiken, die meisten Risiken im Bereich der ungenügend sicher produzierten Anwendungen identifiziert. Zum anderen lassen sich Auswirkungen auf den Support und die Entwicklung von Security Patches. So sind längst bekannte Schwachstellen von Herstellern nach wie vor nicht durch Patches behoben [Sivaraman u. a., 2016].

Weitere Gründe sind in der Beschaffenheit der IoT-Architektur und der verwendeten Protokolle und Technologien zu finden. Grundsätzlich geht man davon aus, dass ein Smart Home vor Angriffen aus der Außenwelt durch den Router inklusive NAT[1] und Firewall, der das Heimnetz mit dem Internet verbindet, geschützt ist [Sivaraman u. a., 2016]. Somit bilden NAT und Firewall klassischerweise die Protect Mechanismen eines Smart Home. Diese Annahme wird jedoch auf verschiedene Art und Weisen untergraben. Zum einen ist im Kommunikationsmodell (c) auf Abbildung 3.2 zu erkennen, dass die Smart Home Anwendungen über einen externen Server gesteuert werden. Das setzt voraus, dass diese im Internet zugänglich sind und damit auch einen Angriffspunkt bieten. Die Kommunikationsmodelle (a) und (b) können zu passiven Risiken werden, indem sogar aus verschlüsseltem Netzwerkverkehr, den die Smart Home Geräte auf externe Server senden, Schlüsse gezogen werden [Apthorpe u. a., 2017a] [Jacobsson u. a., 2014]. Zum anderen werden insbesondere durch neue Protokolle wie das weit verbreitete

[1] Network Address Translation (NAT) ermöglicht es Computern eines LAN über einen gemeinsamen Zugang mit dem Internet zu kommunizieren. Dazu ordnet das NAT interne IP-Adressen eines LAN der öffentlichen WAN-Adresse zu. (https://www.itwissen.info/NAT-network-address-translation-NAT-Verfahren.html)

ZigBee Protokoll, Angriffsflächen frei, die es Angreifern ermöglichen, den Weg über den Router zu umgehen, da diese Protokolle Daten nicht per TCP/IP über den Router transportieren. Meist unterliegen diese Protokolle auch keinen strikten Security Standards. Außerdem sind in der Praxis entdeckte Schwachstellen nur schwer und teuer zu beheben [Müller u. a., 2016]. Das führt bspw. dazu, dass IoT-Objekte auf direktem Wege manipuliert werden können, ohne dass der Router, das Gateway oder die Anwendung etwas davon mitbekommen [Ronen u. a., 2017]. Damit wäre eine SIEM Analyse durch das Sammeln der Logs der klassischen Protect Mechanismen, wie in Kapitel 2 beschrieben, umgangen. Die heterogene Verwendung dieser Protokolle verkompliziert die Situation aus Security-Sicht weiter. Das Einbeziehen der neuen Protokolle ist für einen Smart Home SIEM Ansatz damit unabdingbar.
Die Charakteristiken der IoT-Objekte wie der notwendigerweise niedrige Energiebedarf, die geringen Hardwareressourcen und die Verschlossenheit der Systeme führen zum einen zu einer weiteren Verschärfung der Patchproblematik [Wendzel, 2016]. Zum anderen wird eine Integration von Third-Party-Tools als Protect Maßnahme unmöglich [Lavrova und Pechenkin, 2015].
Nach der Arbeit von Qi Jing et. al mit dem Titel 'Security of the Internet of Things: Perspectives and Challenges' [Jing u. a., 2014] können mögliche Angriffe auf das IoT auf allen Layern der 3-Layer-Architektur erfolgen. Entsprechend wurden in der Arbeit 'Security-Monitoring von Internet-of-Things Endgeräten in lokalen Netzwerken unter Einsatz eines intelligenten Systems' von Fabian Eckhardt [Eckhardt, 2017] beispielhafte Angriffsmöglichkeiten in einer Übersicht auf die drei Layer verteilt (siehe Abbildung 3.3). Diese Übersicht ist für den späteren Verlauf dieser Arbeit hilfreich und zeigt, dass eine gesamtheitliche Detect Maßnahme in Form eines SIEM-Systems alle drei Layer im Smart Home abdecken muss.

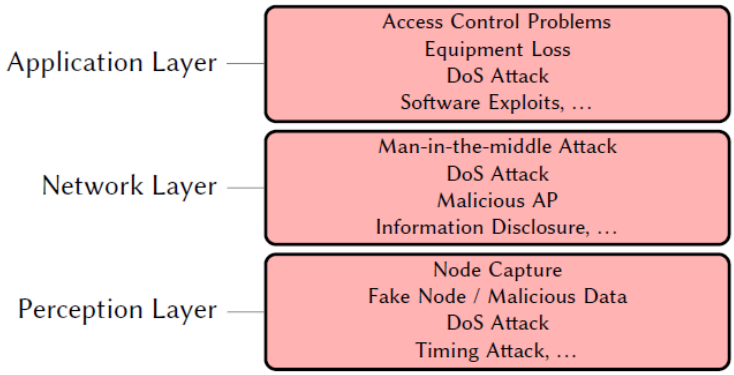


Abbildung 3.3: Angriffsformen in der 3-Layer-Architektur [Eckhardt, 2017]

3.3 SIEM im Smart Home

Dieses Kapitel hat im Abschnitt 3.2 gezeigt, dass ein Smart Home eine Vielzahl von Angriffsflächen bietet, die ausgenutzt werden können und teils hohe Risiken in sich bergen. Auch ist deutlich geworden, dass nach der Überwindung des nativen Sicherheitsmechanismus (Protect) eines Smart Home, dem Router, kaum Sicherheitsbarrieren bestehen. In der Vergangenheit erfolgte Angriffsfälle auf das Smart Home lassen zudem folgern, dass auch hier, ähnlich wie im Unternehmensumfeld oder dem Umfeld kritischer Infrastrukturen, mit einem erfolgreichen Eindringen von Angreifern in das Heimnetz gerechnet werden muss. Die zuvor gezeigte Problematik bei der Integration weitergehender Protect Mechanismen im Smart Home verstärken diese Annahme. Der Schluss daraus muss nach dem Protect-Detect-React Ansatz eine Erweiterung der Detect Maßnahmen sein, wie es in anderen IT-Umfeldern bereits geschieht. Ein möglicher Ansatz ist die in dieser Arbeit behandelte Übertragung eines SIEM-Systems, mit welcher sich dieser Abschnitt auf theoretischer Ebene beschäftigt. Im Folgekapitel 4 wird aus den Erkenntnissen dieses Abschnitts ein theoretischer Übertragungsansatz entworfen, auf dessen Basis im weiteren Verlauf der Arbeit eine konkrete technische Umsetzung erfolgt.

3.3.1 Herausforderungen

Zunächst sollen die bereits abzusehenden Herausforderungen und Schwierigkeiten einer Übertragung des SIEM Ansatzes aus dem Unternehmensumfeld in das Smart Home gezeigt werden. Dabei wird der operative Betrieb eines SIEM-Systems, wie bereits erwähnt, nicht betrachtet, sondern die technische Seite fokussiert. Ein Vergleich mit den Schwierigkeiten eines herkömmlichen SIEM-Systems aus Kapitel 2.2.3 wird gezogen.
Ein klassisches SIEM-System sammelt Log-Daten aus IT-Infrastrukturen, wie auf Abbildung 2.2 zu sehen. Bei den Log-Quellen handelt es sich dabei häufig um standardisierte Produkte, deren Hersteller sich bereits auf eine Anbindung an ein SIEM-System einstellen. Auch kämpfen Hersteller solcher Produkte meist nicht mit den bereits erwähnten Schwierigkeiten der Smart Home Hersteller. Hier existieren meist Wartungsverträge und Hersteller werden neben dem Verkauf des Produktes auch für die Wartung, Lizenz und Laufzeit bezahlt. Dadurch sind bspw. Log-Formate der Produkte bekannt, Log-Einstellungen können vorgenommen und verändert werden. Auch sind die Geräte den Administratoren zugänglich, was eine manuelle Konfiguration, falls nötig, ermöglicht.
Smart Home Komponenten sind dagegen verschlossen und liefern in der Regel keine Logs, was im Abschnitt 3.2 gezeigt wurde. Ein SIEM-System kann jedoch nur mit vertrauenswürdigen und vollständigen Log-Daten zuverlässig arbeiten, was in Kapitel 2 gezeigt wurde. Somit muss ein

Weg gefunden werden, um insbesondere von den Smart Home Komponenten Informationen zu erhalten, die ein SIEM-System verarbeiten kann. Das betrifft vor allem die Anwendungen auf den Gateways und die Anwendungen für die Steuerung durch die Bewohner, aber auch die IoT-Objekte selbst, die in der Regel log-technisch schweigen.
Eine besondere Herausforderung liegt außerdem im Sammeln von Daten der IEEE 802.15.4 Protokolle wie bspw. ZigBee. Da diese nicht über den Heimnetzrouter empfangen und versendet werden und die Gateways der Hersteller verschlossen sind, muss auch hier ein Weg gefunden werden, diesen Netzwerkverkehr in ein mögliches SIEM-System einzupflegen.
Auch der Bereich der Perimeter Security (Analog zu Abbildung 2.2), also der Router, muss betrachtet werden. Router für Privathaushalte erzeugen nicht immer die benötigten Logs oder bieten die Möglichkeit, diese weiterzuleiten, um sie in ein SIEM-System zu steuern. Weitere Security-Logs, die ein herkömmliches SIEM-System eigentlich verarbeitet, werden im IoT und damit auch im Smart Home nicht produziert [Zegzhda u. a., 2016].
Damit ist die in Kapitel 2.2.3 beschriebene notwendige Garantie von Verfügbarkeit und Vollständigkeit der Log-Daten eine an der Quelle, also dem Log-Produzenten, noch größere Herausforderung als bei herkömmlichen SIEM-Systemen.
Weitere in Kapitel 2.2.3 skizzierte Probleme herkömmlicher SIEM-Systeme werden bei Smart Home SIEM-Systemen dagegen geschwächt. So ist die Thematik rund um den Datenschutz unter Umständen nicht so brisant wie im Unternehmensumfeld, da hiervon lediglich der Privathaushalt betroffen ist. Das hängt freilich davon ab, von wem ein SIEM-System betrieben wird und wer Zugang zu den Daten hat.
Auch wenn sich diese Arbeit nicht auf den operativen Betrieb eines SIEM-Systems fokussiert muss in diesem Vergleich erwähnt werden, dass die Frage nach dem operativen Betrieb eines Smart Home SIEM-Systems nach ausführlichen weiteren Betrachtungen und entwickelten Ansätzen verlangt, was eine große Herausforderung darstellt.
Mit dem hohen Log-Volumen eines herkömmlichen SIEM-Systems wird ein Smart Home aufgrund der verhältnismäßig überschaubaren Anzahl (Stand 2017) an zu integrierenden IoT-Geräten nicht zu kämpfen haben.
Die in Kapitel 2.2.3 gezeigte Gefahr der Kompromittierung des SIEM-Systems selbst gilt prinzipiell auch im Smart Home. Auch hier könnte das SIEM-System gezielt ausgeschaltet oder ausspioniert werden. Jedoch beschränkt sich die Anzahl der dort gelagerten Daten auf das eigene Heimnetz, weswegen die Auswirkungen etwas weniger drastisch als in großen Infrastrukturen wären.
Neben den erwähnten und erwarteten Problemen bei der Anbindung der Datenquellen wird eine Betrachtung der restlichen in Kapitel 2 skizzierten Aufgaben eines SIEM-Systems notwendig.

Auch hier sind Eigenarten des Smart Home zu erwarten, wie bspw. die vermutlich heterogene Formatierung und Beschaffenheit der Daten, die beim Filtern und in der Normalisierungsphase behandelt werden muss. 'Best practice' Ansätze oder ein Standard Werk wie das im Kapitel 2 verwendete [BSI, 2013] kann hier nicht als Orientierung verwendet werden.

3.3.2 Bereits vorhandene Ansätze - Related Work

In diesem Abschnitt werden bereits vorhanden Ansätze zur Thematik SIEM im Smart Home aufgeführt und betrachtet. Bei den Ansätzen handelt es sich hauptsächlich um Arbeiten, die sich im generellen IoT Umfeld bewegen. Sie ließen sich aber prinzipiell auf das Smart Home übertragen und werden deshalb ebenfalls betrachtet.

Die Arbeit 'Safe Integration of SIEM Systems with Internet of Things: Data Aggregation, Integrity Control, and Bioinspired Safe Routing' von Peter Zegzhda et. al [Zegzhda u. a., 2016] versucht sich an einem SIEM Ansatz für das IoT. Als besondere Herausforderungen werden hier die Minimierung des aufkommenden Datenvolumens und sichere Routing-Mechanismen von IoT-Geräten zum SIEM-System, welche die Integrität und Vollständigkeit der gelieferten Daten garantieren müssen, betrachtet. Dabei liegt der Fokus vor allem auf großen, sich stets dynamisch verändernden ad hoc Netzwerken. Im Smart Home ist diese Situation entschärft. Das erwartete Datenvolumen ist aufgrund der verhältnismäßig überschaubaren Geräteanzahl (Stand 2017) keine Kernherausforderung beim Umsetzen des SIEM Ansatzes. Auch das Routing ist durch die Gegebenheiten des Heimnetzwerkes handhabbar. Auf konkrete Implementierungsansätze eines SIEM-Systems geht die Arbeit von Peter Zegzhda et. al nicht ein.

Einen Teilaspekt eines SIEM-Systems greift Tim Eckhardt in seiner Arbeit 'Security-Monitoring von Internet-of-Things Endgeräten in lokalen Netzwerken unter Einsatz eines intelligenten Systems' [Eckhardt, 2017] auf. So wird hier eine Möglichkeit entworfen, anhand des Sammelns von Kommunikationsdaten eines IoT-Gerätes Angriffe zu erkennen. Dabei werden machine learning Methoden verwendet, welche den Normalzustand des abgeschlossenen Netzwerkes 'lernen', um anschließend Anomalien entdecken zu können. Allerdings wird dabei lediglich der Transport Layer, ohne Einbezug der IEEE 802.15.4 Protokolle, betrachtet. Somit eignet sich dieser Ansatz theoretisch als ein Teilaspekt eines SIEM-Systems.

Waqas Aman und Einar Snekkenes sind mit ihrer Arbeit 'Event Driven Adaptive Security in Internet of Things' [Aman und Snekkenes, 2014] einen deutlichen Schritt in Richtung gesamtheitliches SIEM-System im IoT gegangen. Dabei wird eine Implementierung mithilfe des Open Source Tools OSSIM [AlienVault] vorgeschlagen und skizziert. Fokus liegt auf der

Erstellung von Korrelationsregeln unterschiedlicher Natur, deren Ansätze auch in der hier vorliegenden Arbeit in Kapitel 8 verwendet werden.
Als Ergänzung kann die Arbeit 'Applying Correlation and Regression Analysis to Detect Security Incidents in the Internet of Things' von Daria Lavrova und Alexander Pechenkin [Lavrova und Pechenkin, 2015] gesehen werden. Sie legt den Fokus auf das Erkennen von noch unbekannten Angriffen anhand 'verbotener' Kommunikationsbeziehungen unterschiedlicher IoT-Geräte.
Beide Arbeiten gehen jedoch davon aus, dass IoT-Geräte Ereignis-Logs produzieren, welche vom SIEM-System verwendet werden können. Da dies zumindest in Smart Home Komponenten selten der Fall ist und da beide Arbeiten ebenfalls nicht auf IEEE 802.15.4 Protokolle und die Möglichkeiten zu deren Analyse eingehen, ist auch hier kein ganzheitlicher Lösungsansatz zu finden.
Das Problem der nicht vorhandenen Ereignis-Logs von IoT-Geräten und Anwendungen geht der Ansatz 'Forensic State Acquisition from Internet Of Things (FSAIoT): A general framework and practical approach for IoT forensics through IoT device state acquisition' von Christopher Meffert et. al [Meffert u. a., 2017] an. Dabei gehen die Autoren von der Problematik der Forensik im IoT aus und erzeugen dazu anhand des Einsatzes des Open Source Gateways openHAB[2] Statusmeldungen der IoT-Geräte. Anhand der Statusänderungen lassen sich Security Incidents im Nachgang forensisch untersuchen. Auch wenn dieser Ansatz nicht direkt auf die Entwicklung eines SIEM-Systems abzielt, können die Methoden zur Ereignis-Log Erzeugung von IoT-Geräten adaptiert werden.

[2] https://www.openhab.org/

4 Eigener Ansatz & Vorgehensweise

In diesem Kapitel wird der Ansatz zur Umsetzung eines SIEM-Systems im Smart Home, beruhend auf den Erkenntnissen aus Kapitel 2 und 3, beschrieben.
Diese Arbeit wird nach einem Top-Down Ansatz vorgehen. Dabei stehen zunächst die Angriffe und Angriffsszenarien im Vordergrund, die das zu implementierende SIEM-System erkennen soll. Damit nach passenden Angriffsszenarien gesucht werden kann, muss zunächst der Aufbau einer exemplarischen Versuchsumgebung stattfinden und beschrieben werden (siehe Kapitel 5). Dieser Versuchsaufbau berücksichtigt bereits den in Kapitel 3.3.2 skizzierten Ansatz zur Erzeugung von Ereignis-Logs der IoT-Geräte [Meffert u. a., 2017]. Auch muss das 3-Layer-Modell auf die konkrete Versuchsumgebung übertragen werden.
Die Angriffsszenarien werden anschließend in Kapitel 6 aus bereits bestehender Literatur ausgewählt und beschrieben. Da das Ziel der SIEM-Implementierung eine ganzheitliche Betrachtung des Smart Home ist, wird bei der Auswahl darauf geachtet, dass Angriffe auf alle Layer des 3-Layer-Modells der Versuchsumgebung enthalten sind. Die gewählten Angriffsszenarien werden dann anhand von Kill Chains in ihre einzelnen Angriffsstufen aufgeteilt. An dieser Stelle sollen auch die zur Erkennung der einzelnen Angriffsstufen notwendigen Log-Quellen und -Daten identifiziert werden, damit sie bei der Implementierung des SIEM-Systems gezielt eingebunden werden können (siehe Kapitel 7). Jede einzelne Angriffsstufe führt zu einem Use Case, der in Kapitel 8 technisch umgesetzt wird.
Für die Implementierung werden Anforderungen an das SIEM-System definiert und Technologien zur Umsetzung entsprechend ausgewählt. Die Implementierung inklusive aller in Kapitel 2 aufgeführten Arbeitsprozesse wird in Kapitel 7 umgesetzt und beschrieben. Die Implementierung soll sowohl auf bereits bestehenden Ansätzen als auch auf Eigenentwicklungen basieren.
Für die Entwicklung der Korrelationsregeln (Use Cases) in Kapitel 8 werden unter anderem Methoden aus den bereits in Kapitel 3.3.2 erwähnten Arbeiten verwendet [Lavrova und Pechenkin, 2015] [Aman und Snekkenes, 2014]
Nachdem die Use Cases umgesetzt sind, wird in Kapitel 9 die Funktionalität des SIEM-Systems getestet, indem die ausgewählten Angriffsszenarien nachgestellt oder deren Symptome erzeugt werden. Damit wird ein Auslösen der Use Cases provoziert.

5 Versuchsumgebung

In diesem Kapitel wird der Aufbau der Smart Home Versuchsumgebung beschrieben, in welcher der SIEM Ansatz implementiert werden soll. Dabei werden zunächst Anforderungen an die Versuchsumgebung definiert, um anschließend deren grundsätzlichen Aufbau und Komponenten zu skizzieren und in Beziehung mit der in Kapitel 3 beschriebenen 3-Layer-Architektur zu setzen. Im Rahmen dieses Kapitels werden zusätzlich die Voraussetzungen geschaffen, um die benötigten Geräte an das spätere SIEM-System anzuschließen. Die genaue Konfiguration bzw. die Wege dieses Anschlusses werden bei der konkreten Implementierung in Kapitel 7 beschrieben.

5.1 Anforderungen

Die Versuchsumgebung, in welcher der SIEM Ansatz implementiert werden soll, muss einem repräsentativen Smart Home entsprechen, um zu möglichst aussagekräftigen Erkenntnissen gelangen zu können.
Um ein repräsentatives, standardmäßiges Smart Home aufbauen zu können, muss sich an bereits vorhanden Arbeiten und weit verbreiteten Komponenten orientiert werden. Auch ist es wichtig, dass sich das Smart Home in der 3-Layer-Architektur für IoTs (siehe Kapitel 3) beschreiben lässt. Nur dadurch wird es möglich, durch die im späteren Verlauf ausgewählten Angriffsszenarien und deren Erkennung durch das SIEM-System zu zeigen, dass alle Layer durch den SIEM Ansatz abgedeckt werden können und damit ein ganzheitlicher Ansatz vorliegt.
Ein zusätzliches Kriterium für die Auswahl der Smart Home Komponenten ist neben deren Verbreitung die Verfügbarkeit von Angriffsszenarien auf diese Komponenten in der Literatur. Nur so können angemessene Angriffsszenarien in Kapitel 6 ausgewählt, in Kapitel 8 als konkrete Use Cases implementiert und in Kapitel 9 nachgestellt und damit deren Erkennung getestet werden.
Auch muss der in Kapitel 2.2 beschriebene Ansatz zur Erzeugung von Statusmeldungen der IoT-Geräte bereits berücksichtigt werden, um bei der Implementierung des SIEM-Systems in Kapitel 7 auf diese Daten zugreifen zu können.
Die Position des späteren SIEM-Systems selbst, sowie der Geräte zur Steuerung des Smart

Home, bspw. Computer oder Smartphones, müssen ebenfalls bereits beim Entwurf und Aufbau der Versuchsumgebung berücksichtigt werden.

5.2 Auswahl & Aufbau

In diesem Abschnitt wird die Auswahl der einzelnen Komponenten erläutert, begründet und deren Funktionsweise dargestellt. Dazu werden die Komponenten der Versuchsumgebung nacheinander besprochen und so zu einem Gesamtaufbau, zu sehen auf Abbildung 5.1 in der 3-Layer-Architektur, zusammengefügt. Die Reihenfolge der Komponentenauflistung ist dabei durch Abhängigkeiten der Komponentenauswahl bedingt.

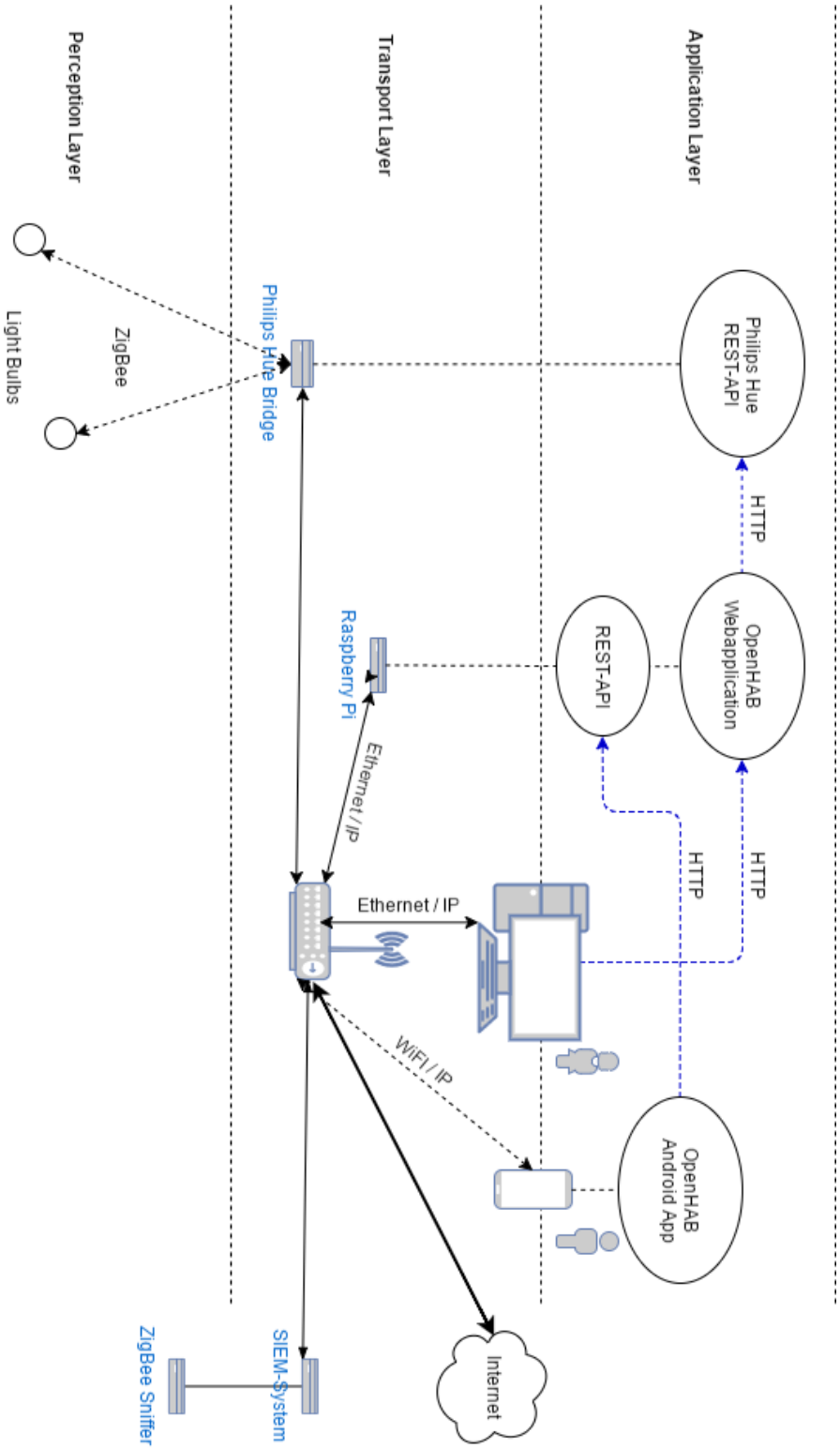


Abbildung 5.1: Versuchsumgebung in der 3-Layer-Architektur (eigene Darstellung)

5.2.1 OpenHAB

Eine der grundlegenden Voraussetzungen für den in Kapitel 2.2 besprochenen, eigenen SIEM Ansatz ist die Verwendung des Ansatzes der dort erwähnten Arbeit von Christopher Meffert et. al [Meffert u. a., 2017]. Hier wird OpenHAB, ein Open Source Tool, welches als zentrale Steuereinheit der vorhandenen Smart Home Komponenten verwendet werden kann, eingesetzt. OpenHAB wird in dieser Versuchsumgebung auf einem Raspberry PI 3 Model B [1] positioniert. Das empfiehlt sich zum einen aufgrund der geringen Kosten eines solches Geräts. Zum anderen ist es so möglich, OpenHAB eine eigene IP-Adresse zuzuweisen, was bei der Analyse der Daten und deren Zuordnung im späteren SIEM-System wichtig wird.
OpenHAB kann durch den Endanwender, also den Smart Home Bewohner, auf zwei unterschiedlichen Wegen gesteuert werden. Zum einen wird eine Webanwendung angeboten, welche im Heimnetzwerk verfügbar ist. Zum anderen stellt OpenHAB eine Android App zur Verfügung, über welche ein Smartphone, sofern es sich im Heimnetz befindet, OpenHAB steuern kann. In dieser Versuchsumgebung werden beide Steuermöglichkeiten bereitgestellt. Angesprochen wird OpenHAB sowohl durch Webanwendung als auch die Android App über eine REST-API, die HTTP-Befehle der Steuergeräte entgegennimmt. Auf Abbildung 5.1 sind diese Steuerverbindungen als blau gestrichelte Pfeile zwischen den Steuergeräten und OpenHAB dargestellt. Dabei sind diese Verbindungen lediglich als logische Verbindungen auf dem Application-Layer zu betrachten, physisch läuft der Datenverkehr über den Router.
OpenHAB unterstützt bereits eine Vielzahl an Smart Home Komponenten unterschiedlicher Hersteller [Meffert u. a., 2017]. Für die Wahl der Smart Home Komponenten wird somit die Unterstützung durch OpenHAB zu einem Auswahlkriterium. Eine unterstützte Komponente kann mit OpenHAB ein sogenanntes 'Binding' erzeugen. Ist ein Binding erzeugt, nimmt OpenHAB Befehle durch die Steuergeräte entgegen (HTTP) und erzeugt anschließend die von der Smart Home Komponente zur Steuerung benötigten Nachrichten. OpenHAB liest gleichzeitig verschiedene Status der Smart Home Komponenten aus. So wird durch Abfragen erkannt, wenn ein Gerät an- oder ausgeschaltet wird, geregelt, entfernt oder hinzugefügt wurde. Damit können unter anderem Änderungen im Perception Layer sichtbar gemacht werden, wenn sich bspw. der Zustand eines Sensors oder Aktors ändert. OpenHAB stellt diese Statusänderungen, wie in der Arbeit von Christoph Meffert et al.[Meffert u. a., 2017] beschrieben, neben eigenen System-Ereignissen als Log-Datei auf dem eigenen Gerät, in diesem Fall dem Raspberry PI, zur Verfügung.
Wie in Abbildung 5.1 zu sehen, befinden sich die Webanwendung, die Android App sowie

[1] https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

die REST-API auf dem Application Layer. Der Netzwerkverkehr, welcher die HTTP-Befehle übermittelt, ist dagegen auf dem Transport Layer einzuordnen.

5.2.2 Philips Hue

Für den Smart Home Versuchsaufbau dieser Arbeit wurde als typische Smart Home Komponente ein Einsteigerset der Lichtsteuerungskomponente Philips Hue [Philips, b] gewählt. Dieses besteht aus einer sogenannten 'Bridge', welche einem in Kapitel 3 beschriebenen Gateway entspricht, und zwei Glühbirnen, die über die Bridge gesteuert werden. Im Folgenden wird diese aufgrund einer Kombination verschiedener Kriterien (siehe u.a. Abschnitt 5.1) getroffene Auswahl begründet.
Eine grundsätzliche Voraussetzung zur Auswahl ist, wie im Abschnitt 5.2.1 erwähnt, die Unterstützung der Komponenten durch OpenHAB. Diese Unterstützung ist im Fall von Philips Hue gegeben [Meffert u. a., 2017].
Eines der Kriterien für einen typischen Smart Home Versuchsaufbau ist die Verbreitung der ausgewählten Komponenten. M. R. Alam et. al zeigen in ihrer Arbeit 'A Review of Smart Homes - Past, Present, Future' [Alam u. a., 2012], dass gesteuerte Beleuchtung in der Kategorie der komfortsteigernden Komponenten stark vertreten ist. Eine Verbreitung der speziellen Komponente dieses Herstellers lässt sich bereits an der Vielzahl anderer Arbeiten erkennen, in welchen sie untersucht oder behandelt wird. Dabei ist eine der hauptsächlichen Anforderungen, dass in bereits bestehender Literatur Angriffsszenarien entworfen wurden, welche die ausgewählte Komponente zum Ziel haben. Die Philips Hue wird bspw. in den Arbeiten 'Smart-Phones attacking Smart-Homes' von Vijay Sivaraman et al. [Sivaraman u. a., 2016], 'An Experimental Study of Security and Privacy Risks with Emerging Household Appliances' von Sukhvir Notra et al. [Notra u. a., 2014] und 'IoT goes Nuclear: Creating a ZigBee Chain Reaction' von Eyal Ronen et al. [Ronen u. a., 2017] behandelt und in skizzierten Angriffsszenarien untersucht. Damit ist eine gute Verbreitungsbasis vorhanden. Auch die von Eyal Ronen et al. geschätzten Verkaufszahlen von Millionen an Philips Hue Geräten im Jahr 2015 deuten auf eine weite Verbreitung hin.
Für die Betrachtung des ganzheitlichen SIEM Ansatzes im Smart Home ist es wichtig, sowohl durch die eingesetzten Komponenten als auch durch die ausgewählten Angriffsszenarien (siehe Kapitel 6) alle Layer der 3-Layer-Architektur abzudecken. Philips Hue verwendet zur Identifikation und Kopplung der Glühbirnen (Aktoren) an die Bridge (Gateway) das ZigBee Light Link Touchlink Commissioning Protocol [Müller u. a., 2016]. Damit befinden sich die Glühbirnen und deren Kommunikation zur Identifikation auf dem Perception Layer (siehe Abbildung 5.1). Zur Steuerung der Glühbirnen wird ebenfalls ZigBee Light Link (ZLL) eingesetzt. Die Philips

Hue Bridge wird über eine REST-API mittels HTTP-Befehlen gesteuert, welche in diesem Versuchsaufbau im Normalbetrieb von OpenHAB erzeugt werden. Damit befindet sich die REST-API der Philips Hue Bridge auf dem Application Layer. Der Netzwerkverkehr wird auch hier vom OpenHAB Raspberry PI über den Router zur Philips Hue Bridge gesteuert. Auch hier ist die blaue, gestrichelte Linie auf Abbildung 5.1 lediglich als logische Verbindung auf dem Application Layer zu betrachten. Damit sind alle Layer der 3-Layer-Architektur involviert.
Neben der Abdeckung aller Layer ist es zusätzlich wichtig, IoT-Protokolle zu betrachten, die eine weite Verbreitung erfahren. Im Fall der Philips Hue ist das ZigBee, welches sich laut unterschiedlicher Arbeiten einer großen Beliebtheit erfreut [Alam u. a., 2012] [Atzori u. a., 2010] (insbesondere auch ZigBee Light Link [Müller u. a., 2016]).
Die Tatsache, dass für den Versuchsaufbau dieser Arbeit lediglich eine Smart Home Komponente ausgewählt wurde, hat unterschiedliche Gründe. Zum einen ist die bereits erfolgte Abdeckung aller drei Layer zu erwähnen. Zum anderen hat die erwähnte Arbeit von Sukhvir Notra et al. [Notra u. a., 2014] gezeigt, dass andere populäre Smart Home Komponenten wie Belkin WeMo und Nest smoke-alarm einer sehr ähnlichen Funktionsweise unterliegen. So sind auch bereits genannte Angriffsszenarien auf andere typische Smart Home Komponenten übertragbar [Sivaraman u. a., 2016]. Auch die von OpenHAB erzeugten Logs würden sich grundsätzlich analog verhalten, sofern die Komponenten von OpenHAB unterstützt werden [Meffert u. a., 2017]. Daraus folgernd ist anzunehmen, dass ein SIEM-System im Smart Home prinzipiell auch andere Komponenten auf ähnliche Art und Weise anbinden könnte wie es in dieser Arbeit mit der Philips Hue erfolgt. Da der hier behandelte SIEM Ansatz zwar ganzheitlich in Bezug auf die Abdeckung der 3-Layer-Architektur sein soll aber in Bezug auf unterschiedlichste Komponenten und Angriffe auf diese nur exemplarisch sein kann, würde durch die Hinzunahme zusätzlicher Komponenten kein entscheidender Mehrwert entstehen. Mit der gleichen Begründung wird im weiteren Verlauf dieser Arbeit lediglich mit einer der beiden zur Verfügung stehenden Glühbirnen gearbeitet. Zukünftige Arbeiten müssen sich dennoch mit weiteren in Kapitel 3 erwähnten Protokollen beschäftigen und diese ebenfalls in den SIEM Ansatz integrieren.

5.2.3 Steuergeräte

Als Steuergeräte, also Geräte, über die der Bewohner OpenHAB und damit die Philips Hue Bridge steuert, wurden hier ein gewöhnlicher Windows-Rechner und ein Android Smartphone gewählt. Über den Windows-Rechner entsteht die Möglichkeit, die OpenHAB Webanwendung zu bedienen, wohingegen das Android Smartphone mit der Android App steuern kann. Auf Abbildung 5.1 sind diese Steuergeräte zu sehen, wobei die durchgezogene schwarze Linie eine

Ethernet-Verbindung vom Windows-Rechner zum Router und die gestrichelte schwarze Linie eine WiFi Verbindung vom Smartphone zum Router darstellt. Die beiden mit HTTP versehenen blauen gestrichelten Linien sind, wie bereits erwähnt, lediglich als logische Verbindungen auf dem Application Layer zu betrachten.

5.2.4 Router

Eine zentrale Instanz eines Heimnetzwerks ist der Router. Wie bereits in Kapitel 3.2 erläutert, bildet er die einzige native Sicherheitsinstanz (Protect Mechanismus), um das Smart Home zu schützen. Wie auf Abbildung 5.1 zu sehen, läuft auch in dieser Versuchsumgebung der gesamte Netzwerkverkehr des Heimnetzes über den Router (bis auf IEEE 802.4.15 Protokolle). So werden bspw. sämtliche HTTP-Befehle zur Steuerung der OpenHAB API durch die Steuergeräte sowie die Steuerung der Philips Hue Bridge durch OpenHAB über den Router geleitet. Der Router befindet sich dabei auf dem Transport Layer der 3-Layer-Architektur.
Aufgrund dieser Tatsachen ist es für den Test eines SIEM-Systems von besonderem Interesse, den Router als Log-Quelle anbinden zu können. Jedoch sind viele handelsübliche Router für Heimnetzwerke, im Gegensatz zu Produkten für die Unternehmenswelt, nicht in der Lage, durch Konfiguration Log-Daten zu erzeugen und bspw. an ein selbst definiertes System weiterzuleiten. Als Lösung für dieses Problem wird in anderen Arbeiten ein selbst konfigurierbarer Router, bspw. auf Basis eines Raspberry PIs eingesetzt [Apthorpe u. a., 2017b]. Die Arbeit von Tim Eckhardt 'Security-Monitoring von Internet-of-Things Endgeräten in lokalen Netzwerken unter Einsatz eines intelligenten Systems' [Eckhardt, 2017] verwendet dazu OpenWRT[2], eine Open Source Firmware, welche unter anderem Konfigurationsmöglichkeiten wie das Weiterleiten von Logs und Daten an andere Geräte ermöglicht.
Dieser Weg wird auch bei dem Versuchsaufbau der vorliegenden Arbeit beschritten. Auf die genaue Konfiguration zum Weiterleiten der Daten wird in Kapitel 7 eingegangen.

5.2.5 SIEM-System

Das SIEM-System selbst, welches in Kapitel 7 implementiert wird, soll auf einem separaten Linux-Rechner laufen, der lediglich eine Ethernet-Verbindung zum Router hat. Alle vom SIEM-System benötigten Daten müssen somit über diese Verbindung zum SIEM-System transportiert werden.

[2] https://openwrt.org/

5.2.6 ZigBee Sniffer

Die für diese Versuchsumgebung ausgewählte Bridge der Philips Hue kommuniziert mit ZigBee, also einem IEEE 802.4.15 Protokoll, mit der Philips Hue Glühbirne. Aufgrund der Tatsache, dass ein gewöhnlicher Router ein IEEE 802.4.15 Protokoll nicht empfangen kann, muss direkt beim Versuchsaufbau eine Maßnahme getroffen werden, um dennoch ein Betrachten des ZigBee Verkehrs zu ermöglichen. Der ZigBee Verkehr ist neben den durch OpenHAB erzeugten Statusänderungen ein entscheidender Bestandteil für die Einbeziehung des Perception Layers. An dieser Stelle wird ein ZigBee Sniffer mit dem Namen 'RZUSBSTICK' vom Hersteller Atmel[3] eingesetzt, welcher in der Lage ist, ZigBee Verkehr zu lesen [Wright, 2009]. Dieser Sniffer hat die Gestalt eines USB-Sticks und wird in diesem Versuchsaufbau direkt an den Linux-Rechner des SIEM-Systems angeschlossen.

5.3 Geräteliste

Die Liste der Geräte Versuchsumgebung wird für die Use Case Entwicklung in Kapitel 6.3 benötigt und ist in Tabelle 5.1 zu sehen. Darauf sind alle relevanten Geräte mit ihren jeweiligen IP- bzw. ZigBee-Adressen abgebildet. Die Spalte LAN/PAN zeigt für alle IP-Geräte das LAN und für die ZigBee-Geräte das PAN indem sie sich befinden. Genauere Erläuterungen zu den Eigenschaften der ZigBee-Geräte wie den Adressen und des PANs folgen im weitere Verlauf dieser Arbeit (siehe Kapitel 6).

	IP-Adresse	ZigBee Adresse	LAN/PAN
Philips Hue Bridge	192.168.1.143	0x0000772c	0x0000117d
Philips Hue Bulb	-	0x0000a02d	0x0000117d
Raspberry PI (OpenHAB)	192.168.1.160	-	192.168.1.0/24
Windows-Rechner	192.168.1.229	-	192.168.1.0/24
Android Smartphone	192.168.1.151	-	192.168.1.0/24
Linux-Rechner (SIEM-System)	192.168.1.189	-	192.168.1.0/24
Router	192.168.1.1 (LAN) 192.168.178.47 (WAN)	-	192.168.1.0/24

Tabelle 5.1: Liste der Geräte in der Smart Home Umgebung

[3] http://www.atmel.com/tools/rzusbstick.aspx

6 Angriffsszenarien

Nachdem in Kapitel 5 die Smart Home Versuchsumgebung dargestellt wurde, wird in diesem Kapitel eine Auswahl von Angriffsszenarien getroffen, welche auf die Versuchsumgebung gefahren werden können. Dabei wird anhand dieser Auswahl gleichzeitig die Zielsetzung an das in Kapitel 7 zu entwickelnde SIEM-System verfeinert. So soll das SIEM-System diese Angriffsformen erkennen. Dazu müssen die Angriffsszenarien so ausgewählt werden, dass sich ihre Angriffsstufen über alle drei Layer der Versuchsumgebung erstrecken (siehe Abbildung 5.1), um die grundlegende Anforderung eines ganzheitlichen Ansatzes prüfen zu können.
Um sich dieser Anforderung anzunähern, ist ein Ziel dieses Kapitels, die Log-Quellen zu identifizieren, die ein implementiertes SIEM-System benötigt, um die ausgewählten Angriffsszenarien erkennen zu können. Anhand dieser Informationen werden in Kapitel 7 die Anbindungen der Log-Quellen an das SIEM-System vorgenommen. Mit Hilfe der verwendeten Log-Quellen kann auch geprüft werden, ob die Auswahl alle drei Layer einbezieht.
Um die benötigten Log-Daten zu identifizieren, müssen die ausgewählten Angriffsszenarien zunächst in ihrer Funktionsweise beschrieben und verstanden werden. Da die ausgewählten Angriffsszenarien mehrstufig durchgeführt werden, werden diese in Kill Chains (Erläuterung von Kill Chains erfolgt in Abschnitt 6.1), also einzelne Angriffsstufen gegliedert. Ziel ist es, im abschließenden Abschnitt dieses Kapitels die einzelnen, relevanten Kill Chain Stufen anhand bewährter Methoden in fachliche Use Cases zu gießen (siehe Kapitel 6.3), welche Blaupausen für die spätere Umsetzung der Use Case Logik (siehe Kapitel 8) sind und eine Identifizierung der jeweils benötigten Daten vornehmen.
Eine tabellarische Zusammenfassung der benötigten Daten sowie eine Prüfung der Abdeckung der drei Layer erfolgt schließlich im Abschnitt 6.3.3.

6.1 Kill Chain

Bevor die Auswahl der Angriffsszenarien und die damit einhergehende Gliederung in Kill Chains erfolgt, ist es vonnöten Kill Chains im Allgemeinen zu erläutern.
Kill Chains beschreiben eine Methode, die ursprünglich entwickelt wurde, um APT-Angriffe in ihren unterschiedlichen Stufen zu bekämpfen. Das SANS Whitepaper von Tony Sager 'Killing

Advanced Threats in Their Tracks: An Intelligent Approach to Attack Prevention' [Sager, 2014] liefert hierzu einen guten Leitfaden.

APT-Angriffe werden zunächst in folgende, vordefinierte Angriffsstufen gegliedert:

1. **Reconnaissance: Gather Information on the target** Ein erstes Auskundschaften des Ziels, was bspw. über öffentlich zugängliche Informationen (Websites, Social Media, etc.) erfolgen kann. Auch bereits die Suche nach bekannten Schwachstellen kann hier stattfinden.

2. **Weaponize the information** Hier wird über die Art des Angriffs entschieden, welcher dann in einen 'auslieferbaren' Zustand gebracht wird (bspw. Malware). Auch die Entscheidung über den 'Lieferweg', sei es über Social Engineering, drive-by Downloads[1] oder andere Methoden, wird hier gefällt und vorbereitet.

3. **Deliver the weapon** An dieser Stelle findet die eigentliche Auslieferung des Angriffs mit dem Ziel statt, ein Zielsystem initial zu infizieren bzw. zu kompromittieren.

4. **Install, spread and hide** Hier findet eine Verbreitung der Infektion oder ein Auskundschaften von internen Netzen statt, nachdem ein System initial infiziert bzw. kompromittiert wurde.

5. **Establish a command and control channel** Um bspw. ausgespähte Daten an den Server des Angreifers zu übermitteln oder eine Möglichkeit einzurichten, infizierte Geräte fernzusteuern, werden möglichst unentdeckt Verbindungskanäle zum Angreifer hergestellt.

6. **Accomblish the task** Hier erfolgt die eigentliche Ausführung des Angriffs über die vorbereiteten Kanäle der Vorstufe.

Bei der Gliederung eines Angriffs in eine Kill Chain ist es nicht entscheidend, dass jede der vordefinierten Stufen befüllt ist, sondern dass alle wirklichen Angriffsschritte in diese eingeordnet werden. Für diese Arbeit werden Kill Chains verwendet, um die in den folgenden Abschnitten ausgewählten und erläuterten Angriffsszenarien in ihre Vorbereitungs- und Angriffsstufen zu gliedern. Da ein SIEM-System einen Angriff oder eine Angriffsvorbereitung möglichst früh aufdecken soll, sollen aus den jeweiligen Kill Chain Stufen Möglichkeiten der Entdeckung abgeleitet werden. Diese werden in Abschnitt 6.3 in theoretischen Use Cases formuliert.

[1] Drive-by Downloads beschreiben eine Methode, Malware an Anwender zu verteilen, indem populäre Websites manipuliert werden. (https://www.security-insider.de/der-drive-by-download-schwer-zu-erkennen-und-brandgefaehrlich-a-306123/)

6.2 Auswahl

Für die Auswahl der Angriffsszenarien sind drei Punkte von entscheidender Bedeutung. Erstens müssen die Angriffsszenarien auf die in der Versuchsumgebung eingesetzten Komponenten passen. Zweitens müssen die Angriffsszenarien alle drei Layer der 3-Layer-Architektur angreifen. Drittens müssen die Angriffe reproduzierbar sein oder deren Symptome so nachgestellt werden können, dass eine Erkennung durch das entwickelte SIEM-System einer Erkennung des echten Angriffs gleich kommt. So werden die Angriffsszenarien beim Testen des SIEM-Systems (siehe Kapitel 9) teilweise nur ansatzweise nachgestellt, was zum einen mit der Komplexität der Szenarien zusammenhängt. Zum anderen würde eine vollständige Durchführung für ein grundlegendes Testen des Ansatzes, auf welchem der Fokus dieser Arbeit liegt, keinen deutlichen Mehrwert bringen. Folglich wird so auch verhindert, dass sich der Fokus dieser Arbeit verschiebt.
Auf Basis dieser Voraussetzungen wurden drei unterschiedliche Angriffsszenarien ausgewählt. Die ersten beiden, Remote Control Attacke auf Hue Bridge und SSDP Reflection Attacke wurden aus den Arbeiten von Sukhvir Notra et al. [Notra u. a., 2014], Vijay Sivaraman et al. [Sivaraman u. a., 2016] und einem akamai Paper [AkamaiTechnologies, 2014] entnommen und kombiniert. Die SSDP Reflection Attacke zielt dabei auf den Transport Layer und die Remote Control Attacke auf Hue Bridge auf den Application Layer.
Inhaltlich und von der Funktionsweise sehr unterschiedlich ist dagegen das dritte Angriffsszenario, ZigBee Take Over Attacke, welches auf den Perception Layer zielt, weder TCP/IP Pakete verwendet noch über Internet-Kommunikation gefahren wird und der Arbeit 'IoT goes Nuclear: Creating a ZigBee Chain Reaction' von Eyal Ronen et al. [Ronen u. a., 2017] entnommen wurde.

6.2.1 Remote Control Attacke auf Hue Bridge

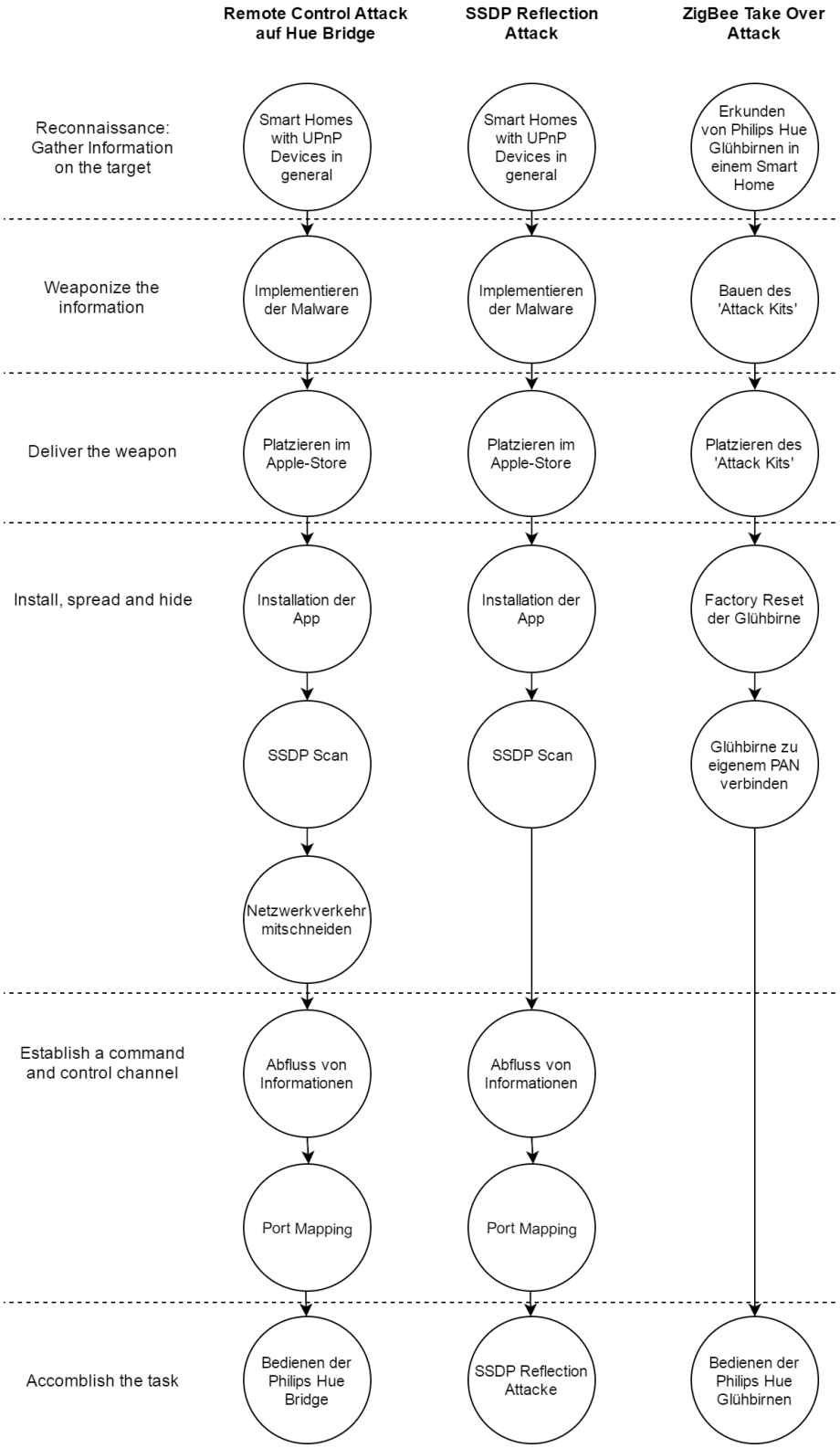
Das Ziel einer Remote Control Attacke ist es, ein Smart Home Gerät von außerhalb des Heimnetzwerkes zu steuern und zu manipulieren, ohne dass der Besitzer (der Bewohner) dies gestattet oder davon in Kenntnis ist. In diesem Fall geht es dabei um die Steuerung der Philips Hue Bridge und damit der Philips Hue Glühbirne.
Da, wie in Kapitel 3.2 beschrieben wurde, ein Heimnetzwerk und damit das Netzwerk eines Smart Home grundsätzlich durch den Router inklusive NAT und Firewall geschützt ist, müssen Angreifer Wege finden, diesen Schutzmechanismus zu umgehen. Das gilt wohlgemerkt nicht für direkte Angriffe auf den Perception Layer, wie in Abschnitt 6.2.3 deutlich wird.
Mit der Überwindung dieser Barriere startet die Arbeit 'Smart-Phones Attacking Smart-Homes' von Vijay Sivaraman et al. [Sivaraman u. a., 2016]. Dazu wird eine Malware unbemerkt über den

Apple AppStore verteilt und so auf einem Apple Smartphone im Heimnetzwerk positioniert. Dass dieser Ansatz auf das in dieser Versuchsumgebung verwendete Android Smartphone übertragbar wäre, zeigt die Argumentation, die Vijay Sivaraman et al. heranziehen. So wird in ihrer Arbeit ein Apple Smartphone verwendet, da es im Android Umfeld noch einfacher wäre, eine solche Schadsoftware zu verteilen. Dazu zeigte ein Nokia Malware Report, dass sich 18 von 20 Smartphone Infektionen über den offiziellen Weg des Google Play Stores verteilt haben. Die vorliegende Arbeit wird nicht näher auf die Entwicklung der hier eingesetzten Malware eingehen, sondern sich auf deren Agieren konzentrieren.
Ist die von Vijay Sivaraman et al. [Sivaraman u. a., 2016] beschriebene Malware erst einmal im Heimnetz angekommen, kann sie zur Ausführung gebracht werden. Die Malware startet im ersten Schritt einen Simple Service Discovery Protocol (SSDP) Scan über das Heimnetz, um Informationen über vorhandene Smart Home Geräte zu sammeln.
SSDP ist ein Protocol, dass zu Universal Plug and Play (UPnP) gehört und der gegenseitigen Identifizierung von Geräten innerhalb eines IP-Netzes dient. SSDP ist in herkömmlichen Smart Home Komponenten verbreitet, sogar in solchen, die keine anderen UPnP Funktionen verwenden. Damit ist es für eine flexible Malware bestens geeignet [Sivaraman u. a., 2016] [AkamaiTechnologies, 2014].
Ein SSDP-Scan wird vom scannenden Gerät an die Multicast Adresse 239.255.255.250 auf Port 1900 gesendet. An SSDP teilnehmende Geräte senden in einer Antwort an Port 1900 ihre IP-Adresse und den Pfad zu einem XML-Dokument, in welchem eine Gerätebeschreibung liegt, zurück. Die von Vijay Sivaraman et al. entworfene Malware sammelt so SSDP Informationen aller teilnehmenden Geräte im Smart Home und sendet sie per HTTP, was in der Regel von Firewalls, Router, etc. nicht geblockt wird, da sonst auch kein Surfen im Internet möglich wäre, an den Server des Angreifers im Internet.
Im zweiten Schritt führt die Malware ein Port Mapping auf dem Routers durch, was bedeutet, dass interne IP-Adressen auf einem Port des Routers von außerhalb erreichbar werden. Dazu ruft die Malware per HTTP die gesammelten IP-Adressinformationen, verpackt in einen UPnP Befehl zum Port Mapping, vom Server des Angreifers ab. Durch das Ausführen dieses UPnP Befehls durch die Malware gegen den Router, werden die Adressen der SSDP Geräte an Ports des Routers gemappt und somit nach außen erreichbar gemacht. Auch diese Nachricht zielt auf Port 1900 des Routers.
Angewandt in der hier aufgebauten Versuchsumgebung würde die Philips Hue, welche ein SSDP Gerät ist (in der API Beschreibung der Philips Hue zu sehen [Philips, a]), auf einen SSDP-Scan reagieren und könnte nach außen hin erreichbar gemacht werden. Besonders problematisch ist es dabei, dass auch die in Kapitel 5 (siehe auch Abbildung 5.1) erläuterte API

der Philips Hue Bridge nach außen erreichbar gemacht werden könnte.
Von diesem Punkt aus könnten bspw. Brute Force Angriffe auf die API geführt werden, um die Zugangsdaten in Erfahrung zu bringen und somit die Beleuchtung direkt zu steuern. Die Arbeit von Sukhvir Notra et al. [Notra u. a., 2014] hat allerdings gezeigt, dass zur Verwendung der Philips Hue API lediglich ein Benutzername notwendig ist, welcher von einem Gerät, das berechtigen Zugriff auf die API hat, im Klartext an diese verschickt wird. Wäre die in der Arbeit von Vijay Sivaraman et al. eingesetzte Malware mit der zusätzlichen Funktion ausgestattet, den HTTP-Verkehr mitzuschneiden und auf dem selben Weg wie die SSDP Informationen an den Angreifer-Server zu senden, könnte der Benutzername ausgelesen werden und der Angreifer wäre in der Lage, die Beleuchtung fernzusteuern, indem er selbst erzeugte HTTP-Nachrichten an den entsprechenden Port des Routers schickt. Das wäre mit Tools wie Android tcpdump[2], welches allerdings ein gerootetes Android voraussetzt, oder tPacketCapture[3], welches ohne root-Berechtigungen auskommt, möglich.
Eine strukturierte Aufgliederung dieses Angriffs anhand der Kill Chain, zu sehen auf Abbildung 6.1 auf der linken Seite, zeigt, dass die ersten drei Stufen außerhalb des Netzwerkes geschehen und damit für ein SIEM-System nicht zu entdecken sind. Die restlichen Stufen finden im Heimnetzwerk statt, können somit für ein SIEM-System relevant sein und werden in Abschnitt 6.3 bewertet und in Use Cases formuliert.

[2] https://www.androidtcpdump.com/
[3] http://www.taosoftware.co.jp/en/android/packetcapture/

Abbildung 6.1: Kill Chains (eigene Darstellung)



6.2.2 SSDP Reflection Attacke

Eine SSDP Reflection Attacke ist eine DDoS-Angriffsform, welche darauf ausgelegt ist, mithilfe fremder, SSDP-fähiger Geräte eine DDoS Attacke gegen ein Zielsystem zu führen [Akamai-Technologies, 2014].
Um diesen Angriff durchzuführen, muss, analog zum Remote Control Angriff, ein SSDP-Scan ein lokales Netzwerk durchsuchen, um Informationen über die Adressen der SSDP-fähigen Geräte zu erlangen und diese nach außen zugänglich zu machen. Dabei sind, wie in der Kill Chain auf Abbildung 6.1 (Mitte) zu sehen, viele der Schritte identisch. So entspricht die Vorbereitungsphase bis zur Installation der App und der Durchführung des SSDP-Scans den Stufen des Remote Control Angriffs. Den Netzwerkverkehr mitzuschneiden ist in diesem Fall nicht notwendig, es genügen die Informationen, die ein SSDP-fähiges Gerät als Antwort auf den SSDP-Scan liefert. Auch die Phasen des Abflusses von Informationen und des Port Mappings auf dem Router sind analog zum Remote Control Angriff zu sehen.
Der letzte Punkt, die Ausführung des eigentlichen Angriffs, unterscheidet sich natürlich. Statt auf die API der Anwendung zuzugreifen, um das Gerät fernzusteuern, wird ein SOAP-Request[4] an Port 1900 der nun bekannten SSDP-fähigen Geräte gesendet, welches wiederum eine Antwort dieser Geräte, erneut über Port 1900, an ein im SOAP-Request angegebenes Gerät provoziert. Beim Senden dieser Antwort ist die XML-Datei, welche das SSDP-Gerät beschreibt (siehe 6.2.1), enthalten. Für den Erfolg eines solchen Angriffs ist die Menge der erspähten UPnP-Geräte und die Größe ihrer beschreibenden XML-Dateien entscheidend.
Dieser Angriff kann ein Smart Home auf zwei Wege treffen. Zum einen können SSDP-fähige Geräte im Smart Home kompromittiert werden, um Attacken auf Dritte zu fahren (wie bereits beschrieben). Zum anderen könnte ein Smart Home auch Opfer eines solchen Angriffs werden. Analog zur Remote Control Attacke auf die Philips Hue Bridge befinden sich die ersten drei Stufen der Kill Chain, zu sehen auf Abbildung 6.1, außerhalb des Heimnetzwerkes und können daher von einem darin platzierten SIEM-System nicht erkannt werden. Die letzten drei Stufen werden bei der Definition der Use Cases, siehe Abschnitt 6.3 betrachtet.

6.2.3 ZigBee Take Over Attacke

Die ZigBee Take Over Attacke unterscheidet sich grundlegend von den bereits erläuterten Angriffsszenarien. Das liegt hauptsächlich daran, dass es sich hierbei um einen sogenannten Seitenkanalangriff handelt, also einen Angriff, der nicht über die üblichen Wege (TCP/IP, Internet) an sein Ziel gelangt [Ronen u. a., 2017]. Die grundlegende Sicherung eines Smart

[4] Ein Simple Object Access Protocol Request (SOAP) wird von UPnP verwendet, um Anfragen an UPnP Geräte zu schicken (bspw. für den SSDP-Scan) [AkamaiTechnologies, 2014].

Home, der Router inklusive Firewall und NAT, wird hierbei umgangen. Dieser Angriff zielt direkt auf die Philips Hue Glühbirne, ohne über den Router an die Philips Hue Bridge zu müssen und greift damit direkt den Perception Layer des Smart Home an. Dabei wird eine Schwachstelle im ZigBee Light Link Protokoll (ZLL) genutzt, um die Glühbirnen vom Personal Area Network (PAN) mit der Philips Hue Bridge zu lösen und anschließend zu einem eigenen PAN zu verbinden. Daraufhin können die Glühbirnen direkt per ZLL vom Angreifer gesteuert werden.
Um diesen Angriff genauer verstehen und im Anschluss dessen Symptome für die Entwicklung der Use Cases herausarbeiten zu können, muss zunächst ein Blick auf die normale Funktionsweise von ZLL im Zusammenhang mit Philips Hue geworfen werden.
Vom ZigBee Light Link Protocol aufgebaute PANs (auch WPANs - Wireless Private Area Networks) beinhalten Teilnehmergeräte mit drei verschiedenen Rollen. Zum einen gibt es die Coordinators, welche das Trust Center des Netzwerks darstellen und für das Management des Netzwerks, inklusive Autorisierung, zuständig sind. Im Fall der Philips Hue stellt die Bridge den Coordinator dar. Zum anderen gibt es ZigBee End Devices, welche die Glühbirne darstellt [Müller u. a., 2016]. Die dritte Art der Teilnehmer, die ZigBee Router, spielen bei der Philips Hue keine Rolle und werden hier nicht weiter betrachtet. Jedes PAN hat seinen eigenen Encryption Key (EK), welcher der Verschlüsselung der Nachrichten innerhalb eines PANs dient. Der EK wird einem neuen Gerät, welches dem PAN beitritt, mit einem weiteren Schlüssel, dem ZLL Master Key[5], verschlüsselt zugeschickt. Der ZLL Master Key ist dabei auf allen ZLL-kompatiblen Geräten herstellerunabhängig hinterlegt.
Soll ein neues Gerät einem PAN hinzugefügt werden, kommt die ZigBee Light Link Touchlink Commissioning Procedure zum Einsatz und folgender Prozess wird in Gang gesetzt (siehe Abbildung 6.2). Ein Initiator (Philips Hue Bridge) sendet Scan Requests als Broadcast auf unterschiedlichen ZigBee Kanälen. In den Scan Requests enthalten ist eine Transaction ID (TrID), bei welcher es sich um eine zufallsgenerierte, 32-Bit lange Zahl handelt, und der Node Type, also der Gerätetyp des Initiators. Das End Device, hier eine Philips Hue Glühbirne, antwortet auf das Scan Request mit einem Scan Response, welches neben dem Gerätetyp und einigen weiteren Informationen ebenfalls die TrID und eine Response ID (RsID), die nach dem gleichen Muster wie die TrID generiert ist, enthält [Müller u. a., 2016]. Die auf Abbildung 6.2 abgebildete Identity procedure ist optional und wird hier nicht weiter behandelt. Nach der erfolgreichen Scan procedure muss der Initiator eine Network join end device procedure starten, um das End Device in das PAN einzugliedern. Dabei wird neben der TrID auch der verschlüsselte EK an das

[5] Der ZLL Master Key wurde 2015 geknackt. Der Key ist für diese Arbeit unbedeutend und wird daher nicht weiter behandelt [Ronen u. a., 2017]

End Device verschickt. Nach einer Bestätigung durch das End Device ist die Prozedur beendet und die Glühbirne kann über die Philips Hue Bridge gesteuert werden. Eine solche Prozedur wird Touchlink Transaction genannt und kann sich eindeutig mithilfe der Kombination aus TrID und RsID identifizieren lassen.
Um zu verhindern, dass sich jeder beliebige Philips Hue Bridge Besitzer fremde Glühbirnen aneignet indem er sie in sein PAN aufnimmt, ist in der ZLL Touchlink Commissioning Procedure ein Mechanismus eingebaut, welcher nur Verbindungen aus nächster Nähe (zwischen 45 cm und 1 m Entfernung) zulässt.
Eyal Ronen et al. haben in ihrer Arbeit 'IoT goes Nuclear: Creating a ZigBee Chain Reaction' [Ronen u. a., 2017] einen Weg gefunden, diese Restriktion zu überwinden. Dazu nutzen sie einen Bug im Code aus, um das End Device, also die Glühbirne, auf Werkszustand zurückzusetzen. Um dieses Zurücksetzen zu erreichen, senden die Autoren einen Network join end device request mit einer zero-TrID an die Glühbirne. Das Protokoll ist an dieser Stelle so implementiert, dass sich die Glühbirnen im Fall einer zero-TrID auf den Werkszustand zurücksetzen und damit alle Informationen über ihr bisheriges PAN verlieren. Zusätzlich haben die Autoren herausgefunden, dass ZLL Geräte wie die Philips Hue Glühbirnen grundsätzlich kompatibel mit anderen ZigBee Netzen sind, die nicht mit dem ZigBee Light Link Protokoll implementiert sind. Ist eine Glühbirne auf Werkszustand zurückgesetzt, beginnt sie automatisch nach ZigBee-Netzen zu suchen, indem sie sogenannte ZigBee beacon requests versendet. Die Autoren reagieren hierauf und senden aktiv ein ZigBee beacon response mit gesetztem 'Association Permit' flag gleich true an die Glühbirne zurück. Die Glühbirne verbindet sich zum PAN der Autoren. Im Gegensatz zum Verbinden zu PANs anderer ZLL Geräte, gibt es bei dieser Interoperabilitätsfunktion zu anderen ZigBee Netzen keine Restriktionen was die Entfernung des Initiators zum End Device angeht. Somit ist es den Autoren gelungen, eine Philips Hue Glühbirne mithilfe der Ausnutzung eines Bugs von ihrem PAN zu lösen, um sie anschließend zu einem ZigBee Netz zu verbinden, ohne die Entfernungsrestriktionen von ZigBee Light Link einhalten zu müssen. Diese Methode ermöglicht eine Verbindung von End Devices zum eigenen PAN mit einer Entfernung von 70 m (Innenräume) bis zu 400 m (Außenbereich). Nach dem erfolgreichen Verbinden der Glühbirne zum eigenen PAN, kann sie problemlos fremdgesteuert werden.
Die Autoren entwickelten zusätzlich einen ZigBee Worm, welcher sich automatisch von Glühbirne zu Glühbirne überträgt, indem ein Firmware Update erzwungen wird. Dieser erweiterte Angriff wird in dieser Arbeit nicht weiter betrachtet, da der Fokus auf der Erkennung eines Angriffs auf den Perception Layer liegt, was mit der ZigBee Take over Attacke bereits gegeben ist.
Betrachtet man die zu diesem Angriff entwickelte Kill Chain (Abbildung 6.1), kann man er-

kennen, dass auch in diesem Fall die ersten drei Angriffsstufen durch ein SIEM-System nicht erkennbar sein können. Damit werden im Abschnitt 6.3 lediglich die Stufen 4, 5 und 6 betrachtet.

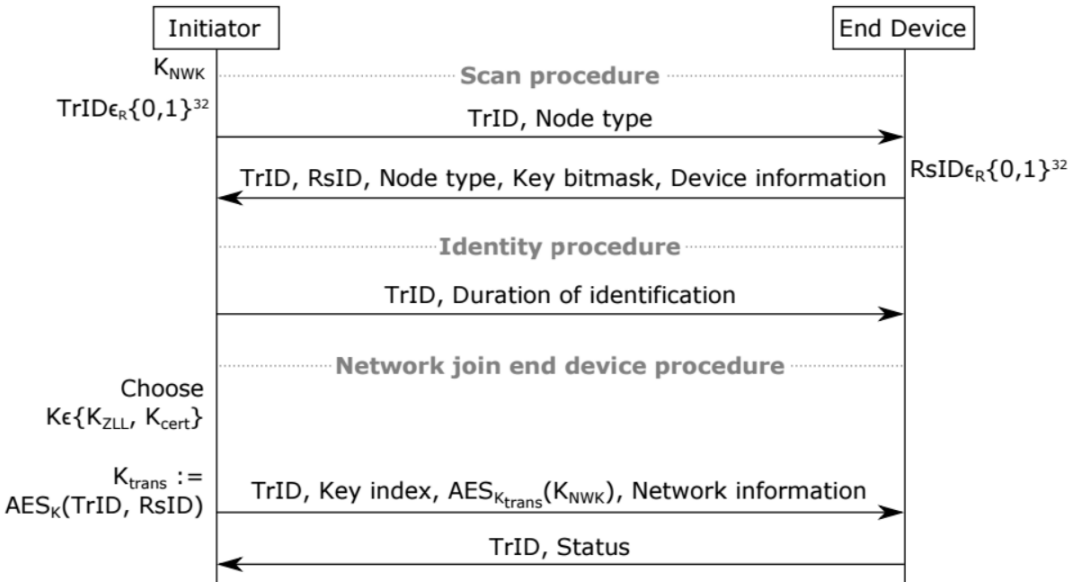


Abbildung 6.2: ZigBee Light Link Touchlink Commissioning Procedure [Müller u. a., 2016]

6.3 Use Case Entwicklung

Die Entwicklung von Use Cases ist, wie in Kapitel 2.2.2 gezeigt, eine der wichtigsten Aufgaben eines SOC beim operativen Betrieb eines SIEM-Systems. Auch wenn sich diese Arbeit auf die technische Seite des SIEM Ansatzes fokussiert, ist es unabdingbar, Use Cases auf Basis der zuvor beschriebenen Angriffsszenarien zu entwerfen, um eine theoretische Grundlage für die technische Implementierung und deren Tests zu haben. Dazu werden die relevanten Angriffsstufen der auf Abbildung 6.1 gezeigten Kill Chains anhand eines vom SANS Institut vorgeschlagenen Frameworks in Use Cases umgesetzt [Gray und Perniola, 2016]. Dabei wurden manche der zu befüllenden Kategorien entfernt, da sie für die Umsetzung dieser Arbeit keine Relevanz haben. Ein Use Case wird im Rahmen dieser Arbeit mit folgenden Informationen ausgestattet:

Description Eine Beschreibung des Use Cases, welche der Beschreibung in der Kill Chain entspricht.

Objective Dieses Feld beinhaltet das Ziel des Use Cases, also die Aufdeckung eines bestimmten Ereignisses.

Threat In diesem Feld wird die Gefahr bzw. das Ziel des Angreifers durch das Auslösen des Ereignisses dargestellt.

Data Requirements An dieser Stelle wird die vom SIEM-System zur Erkennung benötigte Datenquelle angegeben.

Logic Hier wird die Logik der Korrelation textuell dargestellt. Dieses Feld dient in Kombination mit den Data Requirements zur Identifikation der konkret benötigten Daten.

Testing An dieser Stelle wird eine Methode angegeben, durch welche die Angriffsstufe nachgestellt oder deren Symptome erzeugt werden können, um den umgesetzten Use Case zu testen.

Comments / Problems Hier werden notwendige Zusatzkommentare und Probleme bei der Planung bzw. Umsetzung des Use Cases dokumentiert.

Bei der theoretischen Entwicklung der Use Cases wird sich bei einzelnen Stufen herausstellen, dass sie aus verschiedenen Gründen durch ein SIEM-System nicht zu entdecken sind. Diese Use Cases werden dementsprechend aussortiert, in Kapitel 8 nicht technisch implementiert und dementsprechend auch nicht getestet. Da es sich hierbei lediglich um einzelne Stufen

handelt, die Angriffe insgesamt aber noch erkannt werden können, sind die Anforderungen durch das Wegbrechen einzelner Stufen nicht verfehlt.
Ein schwieriger Spagat bei der Entwicklung von Use Cases ist es, diese weder zu präzise noch zu ungenau zu designen. Wird die Logik der Use Cases zu grob gefasst, läuft das System und damit das SOC Gefahr, in False Positives zu ertrinken. Wird sie zu präzise definiert, können leicht abgewandelte Formen der Ereignisse nicht mehr erkannt werden und bleiben somit unentdeckt.
Auch die Angabe der benötigten Daten kann an dieser Stelle noch nicht mit vollständiger Präzision erfolgen. Der hier verfolgte Top-Down Ansatz (Anbindung von Log-Quellen anhand der benötigten Daten der Use Cases) kennt die später gesammelten Daten noch nicht im Detail und kann die entsprechenden Anforderungen lediglich grob erfassen. In Abschnitt 6.3.3 werden die benötigten Daten als Zusammenfassung und Hilfe für die spätere Anbindung der Log-Quellen tabellarisch aufgeführt.
Die Anbindung der Log-Quellen sowie die Beschreibung der implementierten Use Case Logik (Korrelationen) und die Ausführung von Tests wird in den Kapiteln 7, 8 und 9 geleistet.

6.3.1 Remote Control Attacke auf Hue Bridge & SSDP Reflection Attacke

Aufgrund der bereits in den Kill Chains (siehe Abbildung 6.1) erkannten starken Verwandtschaft der beiden Angriffsszenarien Remote Control Attacke auf Hue Bridge und SSDP Reflection Attacke, werden sie in diesem Abschnitt zusammen behandelt.
Die erste Stufe im relevanten Bereich der Kill Chain, die Installation der App, bildet auch den ersten Use Case (siehe Tabelle 6.1). Darin geht es darum, Malware zu erkennen, die auf einem Endgerät installiert wurde, welches sich innerhalb des Smart Home Netzwerkes befindet. Bedauerlicherweise ist es Protect Mechanismen wie Antivirenprogrammen und Firewalls nicht möglich, Malware wie in [Sivaraman u. a., 2016] entwickelt zu erkennen, da sie noch nicht als Malware bekannt ist und 'legal' über einen App-Store oder andere Quellen heruntergeladen wird. Damit wird es mit dem in dieser Arbeit zu implementierenden SIEM-System nicht möglich sein, diesen Use Case umzusetzen.
Die zweite Stufe in der Kill Chain bildet für beide Angriffsszenarien der von der Malware ausgeführte SSDP-Scan und wird damit zu Use Case mit der ID 2. Hier ist das Ziel, zu erkennen, wenn ein SSDP-Scan von einem Gerät im Heimnetzwerk erfolgt, von dem es nicht erwartet wird, da es selbst kein UPnP Gerät bzw. keine Smart Home Komponente ist. Benötigt wird hier zum einen die Geräteliste (Tabelle 5.1). Zusätzlich muss festgelegt werden, von welchen Geräten SSDP-Scans erwartet werden und von welchen nicht. So soll der Scan mit diesen Informationen in Kombination mit dem vom SIEM-System gesammelten Netzwerkverkehr erkannt werden.

Dazu muss nach Netzwerkverkehr von einem als nicht-SSDP Gerät gekennzeichneten Gerät zum Port 1900 der Broadcastadresse 239.255.255.250 gesucht werden (siehe Abschnitt 6.2.1). Ein zusätzliches Suchkriterium muss die Antwort der vom SSDP-Scan gefundenen Geräte sein, welche ebenfalls mit Nachrichten auf Port 1900 antworten. Um diesen Fall zu testen, kann ein SSDP-Scan von einem als nicht-SSDP gekennzeichneten Gerät im Heimnetzwerk ausgeführt werden.
Der Use Case mit der ID 3 kann ebenfalls nicht vom SIEM-System erkannt werden. Dabei handelt es sich um die Kill Chain Stufe 'Netzwerkverkehr mitschneiden'. Ein auf einem Endgerät ausgeführter Prozess, wie bspw. tcpdump (siehe Abschnitt 6.2.1), kann von einem SIEM-System nicht erkannt werden, da er keine Spuren in den üblicherweise verwendeten Daten eines SIEM-Systems hinterlässt. Hierzu wäre eine Überwachung der Prozesse auf den Endgeräten notwendig. Damit wird dieser Use Case ebenfalls nicht umgesetzt.
Da Netzwerkverkehr zur Außenwelt (Use Case ID 4), insbesondere HTTP-Verkehr, grundsätzlich erlaubt ist, ist der Abfluss von Informationen per HTTP schwer zu erkennen. Hier gibt es die Möglichkeit, eine Blacklist von IP-Adressen oder Domänen zu verwenden, bspw. über einen Threat Intelligence Feed (siehe Kapitel 2.2) bezogen, welche kritische oder gefährliche Adressen enthält. Diese Blacklist im Abgleich mit vom SIEM-System gesammeltem Netzwerkverkehr kann einen Abfluss an Daten erkennen. Natürlich ist dafür die Voraussetzung, dass der Server des Angreifers bereits bekannt und in der Blacklist eingepflegt ist. Damit ist dieser Use Case eher unsicher, kann aber dennoch umgesetzt werden. Als Test kann ein auf der Blacklist vorhandener Server von einem Gerät innerhalb des Netzwerkes per HTTP angesprochen werden.
Der nächste Use Case (ID 5) behandelt das Port Mapping. Ziel ist es, ein Notable Event zu erzeugen, wenn ein Gerät ein UPnP Port Mapping auf dem Router des Heimnetzwerkes durchführt. Verwendet werden muss auch hier die Geräteliste, auf welcher der Router gekennzeichnet ist. Fällt nun im Netzwerkverkehr auf, dass ein nicht-SSDP fähiges Gerät Nachrichten auf den Port 1900 des Routers sendet, könnte dies ein Hinweis auf ein durchgeführtes Port Mapping sein. Zum Testen dieses Use Cases kann eine Nachricht von einem nicht-SSDP fähigen Gerät auf Port 1900 des Routers gesendet werden.
Beim Bedienen der Philips Hue Bridge werden zwei unterschiedliche Use Cases benötigt. Als erstes der Use Case mit der ID 6 (siehe Tabelle 6.2), bei dem zum ersten Mal die OpenHAB Status Events zum Einsatz kommen. Dabei soll erkannt werden, wenn ein Benutzer erfolglos versucht, die API der Philips Hue Bridge von einer externen Adresse anzusprechen und die Bridge damit zu steuern. Dazu müssen HTTP-Nachrichten gefunden werden, welche von einer Adresse außerhalb des Heimnetzes kommen und versuchen, auf die API der Philips Hue Bridge

zuzugreifen. Die Philips Hue lehnt diesen Versuch jedoch als nicht autorisiert ab, was in der Antwort auf die HTTP-Nachricht zu erkennen ist. In hoher Anzahl innerhalb kurzer Zeit auftretend, können diese Versuche als Brute Force Angriff gewertet werden.
Wird auf die HTTP-Nachricht jedoch eine erfolgreiche Antwort entdeckt und meldet OpenHAB eine Statusänderung der Philips Hue Bridge, wurde diese erfolgreich angesprochen (Use Case ID 7) und steht damit unter Kontrolle des Angreifers. Als Test kann je eine manuell aufgebaute HTTP-Nachricht mit falschem bzw. richtigem Benutzernamen an die API gesendet werden. Das Testgerät muss sich dabei außerhalb des Heimnetzes befinden.
Der Use Case mit der ID 8 zur Erkennung der SSDP Reflection Attacke soll anschlagen, wenn sich ein Angreifer SSDP-fähige Geräte des eigenen Smart Home zunutze macht, um einen DDoS-Angriff auf ein externes Ziel zu führen. Hier reicht es nicht aus, lediglich anhand einer Korrelation der Geräteliste mit dem Netzwerkverkehr Verbindungen nach außen zu erkennen, da bspw. Noah Apthorpe et al. in ihrer Arbeit 'A Smart Home is no Castle: Privacy Vulnerabilities of Encrypted IoT Traffic' [Apthorpe u. a., 2017b] gezeigt haben, dass Smart Home Geräte regelmäßig nach außen kommunizieren. Da SSDP jedoch grundsätzlich nur im LAN verwendet wird [AkamaiTechnologies, 2014], ist es bereits auffällig, wenn ein SSDP-fähiges Gerät auf der Geräteliste eine externe Adresse auf Port 1900 anspricht. Zusätzlich kann geprüft werden, ob ungewöhnlich viele Nachrichten an Port 1900 eines beliebigen Gerätes gesendet werden, um somit zu erkennen, dass ein Gerät innerhalb des Smart Home das Ziel eines solchen Angriffs ist. Dazu muss eine Korrelation von Ereignissen über einen definierten Zeitraum vorgenommen werden. Zum Testen kann ein beliebiges Testgerät eine Nachricht an ein weiteres Testgerät außerhalb des LAN auf Port 1900 senden. Zum Testen der Alternative kann ein Gerät innerhalb des Netzwerkes eine ungewöhnlich hohe Anzahl an SSDP-Scans starten, um eine ebenso hohe Anzahl an SSDP-Nachrichten zu erzeugen.

ID	Description	Objective	Threat	Data Requirements	Logic	Testing	Comments/Problems
1	Installation der App	Benachrichtigung, wenn sich ein Gerät im Heimnetz mit der Malware infiziert hat	Der Angreifer infiziert ein Gerät mit Malware, um in das LAN zu gelangen und weitere Aktionen ausführen zu können				Eine neue Malware wird von Protect Mechanismen nicht erkannt
2	SSDP Scan	Benachrichtigung, wenn ein SSDP Scan von einem unerwartetem Gerät erfolgt ist	Der Angreifer sammelt Informationen über Geräte im LAN, um diese für weitere Aktionen verwenden zu können	Geräteliste Netzwerkverkehr	Sendet ein unerwartetes Gerät eine Nachricht an Broadcast 239.255.255.250, Port 1900 und Ein IoT-Gerät sendet Nachrichten auf Port 1900 eines unerwarteten Geräts zurück, wird Notable Event erzeugt	Manuelles Ausführen eines SSDP-Scans von einem unerwarteten Gerät	
3	Netzwerkverkehr mitschneiden	Benachrichtigung, wenn eine Malware den Netzwerkverkehr auf infiziertem Gerät mitschneidet	Der Angreifer sammelt HTTP-Nachrichten an Smart Home Geräte aus dem Netzwerkverkehr, um daraus Ableitungen über Credentials und API des Gerätes zu gewinnen	Prozesse auf infiziertem Gerät	Öffnet sich ein Tcpdump-Prozess auf dem infizierten Gerät, wird eine Notable Event erzeugt	tcpdump auf einem Gerät manuell starten	Prozessüberwachung auf Endgeräten ist typischerweise nicht Bestandteil eines SIEM-System
4	Abfluss von Informationen	Benachrichtigen, wenn unerwartetes Gerät Daten an externen Server des Angreifers sendet	Die Malware sendet gesammelte Daten an einen externen Server des Angreifers, welcher diese für weitere Angriffsschritte verwendet	Blacklist Netzwerkverkehr	Baut ein Gerät eine Verbindung zu einem externen Server auf Blacklist auf, wird ein Notable Event erzeugt	Manueller Verbindungsaufbau mit Server auf Blacklist	Unsicherer Use Case, da lediglich Server auf der Blacklist betrachtet werden
5	Port Mapping	Benachrichtigen, wenn ein unerwartetes Gerät ein UPnP Port Mapping auf dem Router vornimmt	Der Angreifer mappt Adressen und Ports von UPnP Geräten auf einen Port des Routers, um die Geräte aus dem Internet erreichbar zu machen und weitere Aktionen durchführen zu können	Geräteliste Netzwerkverkehr	Sendet ein unerwartetes Gerät eine Nachricht an die Adresse des Routers auf Port 1900, wird Notable Event erzeugt	Manuelles Senden einer Nachricht von einem unerwarteten Gerät an Port 1900 des Routers	

Tabelle 6.1: Use Cases: SSDP Reflection Attacke & Remote Control Attacke auf Hue Bridge, Teil 1

ID	Description	Objective	Threat	Data Requirements	Logic	Testing	Comments/Problems
6	Fehlgeschlagenes Bedienen der Philips Hue Bridge	Benachrichtigen, wenn die Philips Hue Bridge von externen Adressen angesteuert wurde	Der Angreifer steuert die Philips Hue Bridge von außen an, und versucht sie zu steuern	Geräteliste Netzwerkverkehr OpenHAB Status Events	Sendet ein externes Gerät eine nicht-autorisierte HTTP-Nachricht an die API der Philips Hue Bridge, wird ein Notable Event erzeugt	Senden einer HTTP-Nachricht von außerhalb des LAN mit falschem Benutzernamen an die API der Hue Bridge	
7	Erfolgreiches Bedienen der Philips Hue Bridge	Benachrichtigen, wenn die Hue Bridge von externen Adressen angesteuert und erfolgreich bedient wurde	Der Angreifer kann die Philips Hue Bridge von außen steuern	Geräteliste Netzwerkverkehr OpenHAB Status Events	Sendet ein externes Gerät eine autorisierte HTTP-Nachricht an die API der Philips Hue und meldet OpenHAB eine Statusänderung der Philips Hue, wird ein Notable Event erzeugt	Senden einer HTTP-Nachricht mit richtigem Benutzernamen von außerhalb des LAN an die API der Hue Bridge	
8	SSDP Reflection Attacke	Benachrichtigen, wenn sich SSDP-fähige Geräte an einer DDoS-Attacke gegen externe Server beteiligen	Der Angreifer bringt die SSDP-fähigen Geräte dazu, SSDP-Nachrichten inklusive ihrer Geräteinformationen an externe Geräte zu schicken und somit eine DDoS-Attacke durchzuführen	Geräteliste Netzwerkverkehr	Geräte senden Nachrichten an Port 1900 externer Geräte und Geräte senden ungewöhnlich viele Nachrichten an Port 1900 eines Geräts	Manuelles Senden einer Nachricht an Port 1900 eines externen Testgeräts oder Manuelles Senden einer Vielzahl von Nachrichten an Port 1900 eines internen Gerätes	

Tabelle 6.2: Use Cases: SSDP Reflection Attacke & Remote Control Attacke auf Hue Bridge, Teil 2

6.3.2 ZigBee Take Over Attacke

Der erste Use Case (siehe Tabelle 6.3) der ZigBee Take Over Attacke hat zum Ziel, den Factory Reset der Glühbirne zu erkennen, welche dadurch aus dem PAN des Smart Home bzw. der Philips Hue Bridge gelöst wird (siehe Kapitel 6.2.3). Zur Erkennung wird eine Kombination aus drei Suchkriterien verwendet. Erstens muss OpenHAB eine Nachricht über die Trennung der Verbindung zur Glühbirne liefern. Zweitens muss nach ZigBee-Nachrichten gesucht werden, die von der Glühbirne gesendet werden, obwohl sie getrennt ist. Ist das der Fall, ist sie nicht offline, also ausgeschaltet, sondern womöglich lediglich vom eigenen PAN getrennt worden. Dadurch kann OpenHAB sie nicht mehr 'sehen' und meldet sie als offline. Drittens muss sichergestellt werden, dass von OpenHAB in der Zwischenzeit keine Nachricht generiert wird, die die erneute Verbindung der Glühbirne markiert. Um diesen Use Case zu testen, kann ein manueller Factory Reset der Glühbirne durchgeführt werden.
Der zweite Use Case dieses Angriffsszenarios soll aufdecken, wenn sich die Glühbirne zu einem fremden PAN verbindet bzw. verbunden hat. Dazu muss im ZigBee-Netzwerkverkehr nach Nachrichten gesucht werden, welche von der Glühbirne in ein fremdes PAN verschickt werden. Zusätzlich muss nach Nachrichten gesucht werden, die von einem fremden Gerät an die Glühbirne geschickt werden, wobei diese bereits als in einem fremden PAN befindlich adressiert wird. Beide Fälle zeigen, dass die Glühbirne übernommen wurde und unter Kontrolle eines Dritten steht. Zum Test dieses Use Cases kann der erfolgreiche Verbindungsaufbau durch eine zweite Philips Hue Bridge durchgeführt werden.
Im dritten Use Case geht es darum zu erkennen, wenn die Glühbirne erfolgreich fremdgesteuert wird. Dazu müsste eine Statusänderung der Glühbirne erkannt werden. Das ist jedoch nicht möglich, da sich die Glühbirne, sobald sie sich in einem fremden PAN befindet, nicht mehr durch OpenHAB überwachen lässt. Somit kann dieser Use Case nicht implementiert werden. Da die Voraussetzungen zur Steuerung der Glühbirne von Seiten des Angreifers jedoch bereits im Zusammenhang mit dem vorherigen Use Case geschaffen werden und eine Fremdsteuerung bereits durch das Einbinden in ein fremdes PAN erfolgt ist, ist das Fehlen dieses letzten Use Cases kein großer Verlust an Erkennungsmöglichkeiten durch das SIEM-System.

ID	Description	Objective	Threat	Data Requirements	Logic	Testing	Comments/Problems
1	Factory Reset der Glühbirnen	Benachrichtigen, wenn ein Factory Reset auf einer Philips Hue Glühbirne durchgeführt wurde	Der Angreifer führt ein Factory Reset der Glühbirne durch, wodurch diese aus dem PAN gelöst wird	Geräteliste ZigBee Netzwerkverkehr OpenHAB Status Events	OpenHAB Status Events melden eine Glühbirne als nicht erreichbar und kein OpenHAB Event mit Reaktivierung der Glühbirne nach der vorherigen Bedingung und die entsprechende Glühbirne sendet weiterhin ZigBee-Nachrichten	Factory Reset einer Glühbirne durchführen	
2	Glühbirne wurde zu fremdem PAN verbunden	Benachrichtigen, wenn eine Philips Hue Glühbirne zu einem anderen PAN verbunden wurde	Der Angreifer verbindet die Glühbirne zu seinem eigenen PAN und hat sie damit unter Kontrolle	Geräteliste ZigBee Netzwerkverkehr	Glühbirne sendet ZigBee-Nachrichten an ein Gerät in einem fremden PAN oder fremdes Gerät sendet Nachrichten an eigene Glühbirne, welche bereits fremdem PAN zugeordnet wird	Verbindung zu PAN einer zweiten Philips Hue Bridge durchführen	
3	Bedienen der Philips Hue Glühbirnen	Benachrichtigen, wenn Philips Hue Glühbirne von einem fremden PAN gesteuert wurde	Der Angreifer steuert die Glühbirnen	Geräteliste ZigBee Netzwerkverkehr OpenHAB Status Events	Glühbirne ändert ihren Status ohne ZigBee-Nachricht von der eigenen Philips Hue Bridge oder HTTP-Nachricht an die Philips Hue Bridge		Statusänderungen der Glühbirnen können nach der Verbindung zu einem fremden PAN nicht mehr erfasst werden

Tabelle 6.3: Use Cases: ZigBee Take Over Attacke

6.3.3 Zusammenfassung der benötigten Daten

Um im nächsten Kapitel (7) bei der technischen Umsetzung des SIEM-Systems gezielt die benötigten Daten in das SIEM-System zu integrieren, wird im Folgenden eine tabellarische Zusammenfassung dieser Daten erstellt (Tabelle 6.4). Dazu werden neben der Aufführung der Datentypen zur Vorbereitung der späteren Normalisierung bereits die benötigten Parameter definiert und mit einheitlichen Namen versehen, die in den Grundsätzen einem Vorschlag von Daria Lavrova et al. folgen, die in ihrer Arbeit 'Applying Correlation and Regression Analysis to Detect Security Incidents in the Internet of Things' [Lavrova und Pechenkin, 2015] Vorschläge zur Normalisierung von Daten im IoT-Umfeld machen.
Der erste zu implementierende Use Case (ID 2 in Kapitel 6.3), benötigt zum einen die Geräteliste (Kapitel 5.3), welche hier nicht weiter betrachtet wird, da sie nicht automatisiert in das SIEM-System einfließen soll, sondern manuell bei der Implementierung der Use Cases zum Tragen kommt. Zum anderen wird Netzwerkverkehr benötigt, um den SSDP Scan zu erkennen. Dieser wird in Flow-Daten zu erkennen sein, da es sich hierbei um UDP-Verbindungen handelt. Damit muss die IP-Adresse des Ziels als 'dest_ip', dessen Port als 'dest_port' und der Typ der Verbindung eines Flow-Events als 'transport' eingebunden werden. Um in der späteren Analyse dieses Use Cases erkennen zu können, von welchen Geräten der Scan erfolgt ist, wird auch die Quell-IP als 'source_ip' und der dazugehörige Port als 'source_port' benötigt. Um Flow-Daten grundsätzlich schnell identifizieren zu können, wird ein typbeschreibendes Feld 'type' benötigt.
Use Case ID 4 benötigt neben der Blacklist, welche exemplarisch und ebenfalls statisch bei der Umsetzung der Use Cases verwendet wird, nicht automatisiert in das System einfließt und damit an dieser Stelle nicht weiter betrachtet werden muss, die Quell- und Ziel-IP sowie die dazugehörigen Ports der HTTP-Daten.
Use Case ID 5 kommt mit den bereits aufgeführten Flow-Daten aus.
Use Case ID 6 benötigt zum einen Netzwerkverkehr in Form von HTTP-Nachrichten. Hier ist neben dem Datentyp als 'type' besonders die Quell-IP als 'source_ip', und deren Port wichtig, um den Host eines Angreifers identifizieren zu können. Zum anderen wird ebenfalls die 'dest_ip' sowie deren Port als 'dest_port' benötigt, um nach HTTP-Nachrichten suchen zu können, die auf die Philips Hue Bridge zielen. Um zu erkennen, ob sich der Angreifer mittels HTTP erfolgreich an der API der Philips Hue Bridge anmelden konnte, muss der Inhalt des HTTP-Requests als 'request' und der Inhalt der HTTP-Response als 'response' eingebunden werden. Auch muss für diesen Use Case erkannt werden, ob sich der Quell-Host außerhalb des Smart Home LAN befindet. Dazu wird das Feld 'requested_host' benötigt, welches zeigen soll, an welche Adresse die HTTP-Nachricht zunächst gesendet wurde. Ist ein Port Mapping auf

dem Router erfolgt und wird die HTTP-Nachricht über diesen Weg in das LAN und auf die Philips Hue Bridge geschleust, ist die erste Ziel-Adresse des HTTP-Requests die WAN-Adresse des Routers.
Um in Use Case ID 7 zu erkennen, dass eine HTTP-Nachricht zu einer physischen Veränderung der Philips Hue Bridge bzw. deren Glühbirne geführt hat, werden hier neben den in Use Case ID 6 identifizierten HTTP-Daten die OpenHAB Status Events benötigt. Um zu erkennen, welche Art der Änderung der Status der Glühbirne erfahren hat, wird die Art des Ereignisses als 'eventtype' eingebunden. Desweiteren wird der Inhalt der Änderung von OpenHAB als 'message' gebraucht, um neben der Art des Events die genaue Änderung in den Use Case integrieren zu können.
Für den Use Case ID 8, den SSDP-Scan, wurden bereits alle notwendigen Daten erwähnt.
Der erste ZigBee Use Case benötigt zum einen ebenfalls die OpenHAB Status Events, kommt aber mit den bereits identifizierten Inhalten aus. Zum anderen wird der ZigBee Netzwerkverkehr benötigt. Dabei muss ebenfalls die Quelle und das Ziel enthalten sein, um im Abgleich mit der Geräteliste identifizieren zu können, ob es sich bei den Sendern oder Empfängern um die eigenen Geräte handelt (der in Kapitel 5 beschriebene ZigBee Sniffer wird auch ZigBee-Netzwerkverkehr von fremden Geräten in der Umgebung ausgeben). Da es sich allerdings um keine IP-Geräte handelt, werden die Adressen als 'dest_addr' und 'source_addr' benötigt.
Für den zweiten ZigBee Use Case ist zusätzlich noch das Ziel-PAN als 'dest_pan' notwendig.

Flow Daten	**HTTP**	**OpenHAB Status Events**	**ZigBee-Daten**
type	type	type	type
source_ip	source_ip	eventtype	source_addr
dest_ip	dest_ip	message	dest_addr
source_port	source_port		dest_pan
dest_port	dest_port		
transport	request		
	response		
	requested_host		

Tabelle 6.4: Zusammenfassung der benötigten Daten

Anhand der hier dargestellten Übersicht über die benötigten Daten kann erkannt werden, dass Daten aller drei IoT-Layer einbezogen werden. So ist in Zusammenhang mit der auf Abbildung 5.1 vorgenommen Einteilung der Versuchsumgebung in die 3-Layer-Architektur zu sehen, dass durch die Verwendung der HTTP-Daten der Application Layer, durch die

Flow-Daten der Transport Layer und durch die Einbindung der ZigBee- sowie der OpenHAB Status Events der Perception Layer abdeckt ist. Damit genügen die ausgewählten Angriffe der Anforderung einer ganzheitlichen Betrachtung des Smart Home.

7 Implementierung

In diesem Kapitel wird die konkrete technische Implementierung des in Kapitel 4 beschriebenen eigenen Ansatzes erläutert. Dazu werden zunächst die funktionalen und technischen Anforderungen an das System in Bezug auf in vorherigen Kapiteln dieser Arbeit bereits aufgeführte Eigenschaften eines SIEM-Systems gesammelt. Darauf folgend wird ein Überblick über die gesamte Architektur des Systems gegeben, um anschließend die einzelnen Komponenten genauer zu beschreiben. Die Beschreibung der Komponenten wird anhand der Schritte der Funktionsweise eines SIEM-Systems (siehe Kapitel 2.2.1) strukturiert. In Abschnitt 7.3 wird dabei neben der Beschreibung der Komponenten zur Datensammlung insbesondere auf das Erschließen der Datenquellen eingegangen.

7.1 Anforderungen an das System

Ziel des implementierten SIEM-Systems ist es, die Tauglichkeit eines SIEM Ansatzes im Smart Home auf technischer Ebene zu zeigen. Dazu sollen die in Kapitel 6 gewählten und erläuterten Angriffsszenarien erkannt werden. Damit wäre gezeigt, dass der SIEM-Ansatz Angriffe auf allen Layern der 3-Layer-Architektur (siehe Kapitel 2) erkennen kann. In Kapitel 6.3 wurden diese Angriffsszenarien in einzelne Use Cases aufgebrochen, die mit jeweils zur Erkennung benötigten Daten versehen wurden. Um im späteren Verlauf der Arbeit, konkret in Kapitel 8, wenn die Use Cases implementiert werden, die Funktionalität des SIEM-Systems zeigen zu können, ist die zentrale Anforderung an das System, die in Kapitel 6.3.3 benötigten Daten zu sammeln und zu verarbeiten.
Um sich der technischen Lösung des Smart Home SIEM-Systems zu nähern, müssen zunächst weitere Anforderungen gesammelt werden, um anschließend eine passende Auswahl an Komponenten und Technologien treffen zu können.
Eine grundlegende Anforderung an das System ist, dass es in der Lage ist, eine große Anzahl an unstrukturierten, mitunter unerwarteten Daten sammeln und speichern zu können. Eine schnelle Verarbeitung der Daten und die Möglichkeit, diese schnell durchsuchen zu können sind für ein SIEM-System essentiell, da nur so Korrelationen zeitnah ausgeführt werden und Ergebnisse liefern können (siehe Kapitel 2). Um dieses Ziel erreichen zu können, muss eine

entsprechende Datenbanktechnologie gewählt werden. Aufgrund der unter Umständen schnell wachsenden Datenmengen muss ebenfalls Wert auf mögliche Skalierbarkeit und Verteilbarkeit des Systems gelegt werden, auch wenn die Datenmenge gegenüber einem herkömmlichen SIEM-System vermutlich überschaubar bleibt.
Eine Stufe vor der Speicherung und Verarbeitung der Daten wartet die in Kapitel 3 beschriebene Schwierigkeit der mangelhaften Verfügbarkeit und Beschaffenheit der Log-Daten im Smart Home. Dies ist eine der größten Herausforderungen bei der Umsetzung des Smart Home SIEM-Systems. Hier müssen Wege gefunden werden, diese Daten erheben, sammeln und verarbeiten zu können. Um die gesammelten Daten analysieren zu können, müssen Möglichkeiten zur Normalisierung der Daten jeder Datenquelle geschaffen werden. Auch das Filtern der Daten muss ermöglicht werden, um die Datenmenge reduzieren zu können und anschließend nur die Daten analysieren zu müssen, die zur Erkennung der Use Cases notwendig sind. Aufgrund der Tatsache, dass in Zukunft weitere Angriffsformen erkannt und damit möglicherweise weitere Datenquellen an das System angeschlossen werden müssen, ist Wert auf eine einfache Erweiterbarkeit des Systems zu legen. Aus denselben Gründen müssen Komponenten auf einfachem Wege austauschbar sein, wenn sich bspw. Datenquellen ändern.
Obwohl in dieser Arbeit kein Fokus auf die Umsetzung oder Beschreibung des operativen SIEM-Betriebs gelegt wird (siehe Kapitel 2.2.2), müssen die gesammelten Daten visualisierbar sein, um Benutzern zu ermöglichen, die Daten zu analysieren und auszuwerten. Dies ist auch für die Durchführung der Tests in Kapitel 9 notwendig.
Ebenfalls essentiell für die Funktionsweise des SIEM-Systems ist dessen Fähigkeit, die gesammelten, gefilterten, normalisierten und gespeicherten Daten mithilfe von Korrelationssuchen miteinander in Beziehung zu stellen und damit die in Kapitel 6.1 herausgearbeiteten Angriffsstufen, welche in Kapitel 6.3 in Use Cases gegossen wurden, erkennen zu können. Aus den Ergebnissen der Korrelationen muss das SIEM-System Notable Events erzeugen können, welche wiederum in der Datenbanktechnologie gespeichert werden und dem Benutzer über die bereits erwähnte Visualisierungsmöglichkeit zur Verfügung gestellt werden. Auf Basis dieser Ergebnisse würde in einem SIEM-System im Unternehmensumfeld ein SOC die Priorisierung und Analyse von Security Incidents vornehmen und möglicherweise einen Incident Response Prozess einleiten (siehe Kapitel 2.2.2). Die Möglichkeit der Priorisierung wird bei der technischen Umsetzung innerhalb dieser Arbeit nicht betrachtet, da der Fokus auf der technischen Machbarkeit liegt und die Arbeit eines SOC sowie dessen operative Prozesse nicht tiefer betrachtet werden.
Weitere Anforderungen an das System insgesamt ergeben sich aus den in Kapitel 2.2.3 beschriebenen Gefahren eines SIEM-Systems. Um verlässliche Informationen liefern zu können, muss

bei der Sammlung der Daten Wert auf deren Vollständigkeit und Integrität gelegt werden, da bei fehlerhaften oder fehlenden Daten Angriffe übersehen oder falsch gedeutet werden könnten. Auch eine hohe Verfügbarkeit ist von Bedeutung, da ein SIEM-System selbst zum Angriffspunkt werden könnte und bei einem Ausfall keine Angriffe mehr erkannt werden könnten. Um das SIEM-System vor Angriffen zu schützen, ist eine der grundlegenden Anforderungen, es nur innerhalb des Smart Home LAN sichtbar zu machen und durch Authentifizierungsmaßnahmen zu schützen. Diese Anforderungen erfüllen sich im Smart Home 'von selbst', da der Router, wie in Kapitel 3.2 gezeigt, bereits vor Zugriffen von der Außenwelt schützt, insofern kein Port Mapping auf dem Router geschieht, siehe Use Case ID 5, Kapitel 6.3.
Bei der Auswahl der Komponenten und Technologien ist darauf zu achten, dass diese aufgrund der Rahmenbedingungen dieser Arbeit und des technischen Projekts unter Open Source Lizenzen zu verwenden sind.

7.2 Aufbau

7.2.1 Verwendete Technologien und Architektur

In diesem Abschnitt werden die unterschiedlichen, auf Basis der zuvor beschriebenen Anforderungen ausgewählten Technologien erläutert, um sie anschließend in der Gesamtarchitektur des Systems miteinander in Zusammenhang zu bringen. Dabei werden die Technologien nicht in voller Tiefe beschrieben, da der Fokus dieser Arbeit auf der technischen Übertragung des SIEM Ansatzes insgesamt liegt.
Zunächst ist es wichtig, die Auswahl der Datenbanktechnologie zu treffen, da sie das zentrale Element des Systems bildet. Im Rahmen dieser Arbeit wurde das Open Source Produkt Elasticsearch[1] gewählt. Elasticsearch ist dabei keine herkömmliche Datenbank sondern eine Suchmaschine, die Daten nicht in strukturierten Datenbankeinträgen wie bspw. bei einer relationalen SQL-Datenbank, sondern in einem sogenannten NoSQL-Format, in diesem Fall JSON, speichert [retresco]. Eine solche Lösung ist für ein SIEM-System notwendig, da es sich bei den zu sammelnden Daten um unstrukturierte Daten handelt. Eine weit verbreitete Alternative zu Elasticsearch ist Apache Solr[2]. Gegenüber Apache Solr hat Elasticsearch zum einen den Vorteil, dass es komplett schemalos arbeitet. Somit müssen zu Beginn eines Projektes keine Schemata der erwarteten Daten angegeben werden, was für ein SIEM-System essentiell ist, da die Daten teilweise Felder und Formate enthalten, die nicht von Beginn an erwartbar sind. Auch für den möglichen Anschluss zusätzlicher Log-Quellen ist dies ein wichtiges Kriterium. Ein Nachteil

[1] https://www.elastic.co/de/products/elasticsearch
[2] http://lucene.apache.org/solr/

der Schemalosigkeit ist die daraus entstehende Notwendigkeit, Suchen möglichst präzise zu formulieren, da keine automatische Vorsortierung der Daten stattfindet.
Ein weiterer Vorteil von Elasticsearch gegenüber Apache Solr ist die grundsätzliche Ausrichtung auf eine mögliche Verteilung und Skalierung der Datenbasis. Die Skalierbarkeit ist, wie in den Anforderungen gezeigt, eine wichtige Anforderung an ein SIEM-System.
Ein ebenfalls wesentlicher Faktor für die Wahl von Elasticsearch ist die große Bandbreite an unterstützten Komponenten, die sich unter dem Begriff des Elastic Stack[3] sammeln. Dabei sind insbesondere Packetbeat zum Sammeln von Netzwerkverkehr, Logstash als Schnittstelle zum Sammeln, Verarbeiten und Weiterleiten von Daten an andere Komponenten und Kibana zum Visualisieren von gesammelten Daten von Interesse.
Elasticsearch wurde auch in Vorschlägen bereits bestehender Arbeiten in ähnlichem Kontext verwendet. Dabei ist die Arbeit von Tim Eckhardt [Eckhardt, 2017] und der Vorschlag zu einem Eigenbau SIEM von Daniel Mahrenholz [Daniel Mahrenholz, 2016] zu nennen, an dessen Architektur sich der technische Aufbau dieser Arbeit grob orientiert. Damit ist der Einsatz und die Tauglichkeit von Elasticsearch in dem hier untersuchten Kontext auch praktisch bereits erprobt.
Eine zweite grundlegende Technologieentscheidung hängt mit der Wahl der Anwendungsarchitektur zusammen. Für die Umsetzung des Smart Home SIEM-Systems wurde Docker[4] als eine Container-Technologie gewählt, um einzelne Komponenten zu strukturieren und mit dem Tool Docker-Compose[5] zu orchestrieren. Die Idee dabei ist, das SIEM-System mithilfe der Docker-Technologie als microservice-basierte Anwendung zu implementieren. Der Einsatz einer microservice-basierten Architektur hängt mit den geforderten Eigenschaften der Skalierbarkeit, Verfügbarkeit und Erweiterbarkeit zusammen. So ist es in diesem Architekturmodell möglich, Komponenten, also Container, zur gesamten Anwendung nachträglich hinzuzufügen und auszutauschen. Eine wesentliche Eigenschaft eines microservice-basierten Architekturmodells ist auch die Verfügbarkeit, da ein einzelner ausgefallener Container nicht das gesamte System lahmlegt. Skalierbarkeit wird durch die Möglichkeit, Container bei Bedarf zu replizieren und auf unterschiedliche Hosts zu verteilen, ermöglicht [Jäger, 2017]. Diese Eigenschaft harmoniert gut mit der Skalierbarkeit von Elasticsearch.
Um konsequent eine microservice-basierte Architektur umzusetzen, wird in dieser Arbeit, wie bspw. im ebenfalls microservice-basierten MARS-Projekt der HAW Hamburg [Hüning u. a., 2016], eine Open Source Middleware in Form von RabbitMQ[6] eingesetzt. RabbitMQ soll in die-

[3] https://www.elastic.co/guide/en/elastic-stack/current/elastic-stack.html
[4] https://www.docker.com/
[5] https://docs.docker.com/compose/
[6] https://www.rabbitmq.com/

sem System als Zwischenspeicher für gesammelte Daten dienen, um sie mit unterschiedlichen Komponenten weiterzuverarbeiten und letztlich Elasticsearch zur Verfügung zu stellen. Das hat den Vorteil, dass gesammelte Daten auch dann nicht verloren gehen, wenn Elasticsearch bspw. nicht zur Verfügung steht und dient damit der Anforderung der Datenvollständigkeit. RabbitMQ weißt ebenfalls eine hohe Kompatibilität zu den Komponenten des Elastic Stack auf und wird auch in der nicht-microservice-basierten Anwendung von Daniel Mahrenholz eingesetzt [Daniel Mahrenholz, 2016].
Nach der Auswahl der grundlegenden Technologien und dem microservice-basierten Architekturmodell ist auf Abbildung 7.1 die Architektur des Smart Home SIEM-Systems zu sehen. Dabei handelt es sich bei jedem dargestellten Service um einen eigenen Docker-Container.
Die in Wolkenform dargestellten Komponenten stellen die unterschiedlichen Datenquellen dar. Die Datenquelle des Routers wird dabei von der Komponente packetbeat angezapft, welche den vom Router zur Verfügung gestellten Netzwerkverkehr sammelt und an die nächste Komponente, logstash-network-in leitet. Diese Komponente filtert und normalisiert (nur teilweise, mehr in Abschnitt 7.5) die gesammelten Daten und leitet sie im JSON-Format als sogenannte Events an die RabbitMQ weiter. Analog dazu verhält es sich mit den Komponenten logstash-zigbee-in und logstash-openhab-in. Diese Komponenten werden als Proxy Dienste bezeichnet und werden in Abschnitt 7.3 und 7.5 genauer behandelt.
Die syslog-parser Komponente liest bestimmte Events aus der RabbitMQ ein und bearbeitet diese zur Normalisierung mithilfe eines speziell geschriebenen Python-Skriptes. Über diesen Weg werden lediglich OpenHAB Status Events normalisiert (genauere Beschreibung in Abschnitt 7.5).
Nachdem alle Events normalisiert in der rabbitmq Komponente vorliegen (hierzu erfolgt die genaue Beschreibung in Abschnitt 7.4), werden sie von logstash-events an elasticsearch weitergeleitet und dort abgelegt (Beschreibung in Abschnitt 7.6).
Innerhalb der correlator Komponente laufen die für die Umsetzung der Use Cases entwickelten Korrelationssuchen in Form von Python-Skripten. Diese durchsuchen den Elasticsearch Datenbestand. Bei einem Treffer wird ein Notable Event erzeugt und zurück in die RabbitMQ gelegt, um anschließend über logstash-notable-events in Elasticsearch gespeichert zu werden. Die correlator Komponente wird in Abschnitt 7.7 beschrieben, die darin umgesetzten Use Cases in Kapitel 8.
Die Komponente kibana visualisiert sowohl die gesammelten Events als auch die erzeugten Notable Events und wird im Abschnitt 7.8 erläutert.

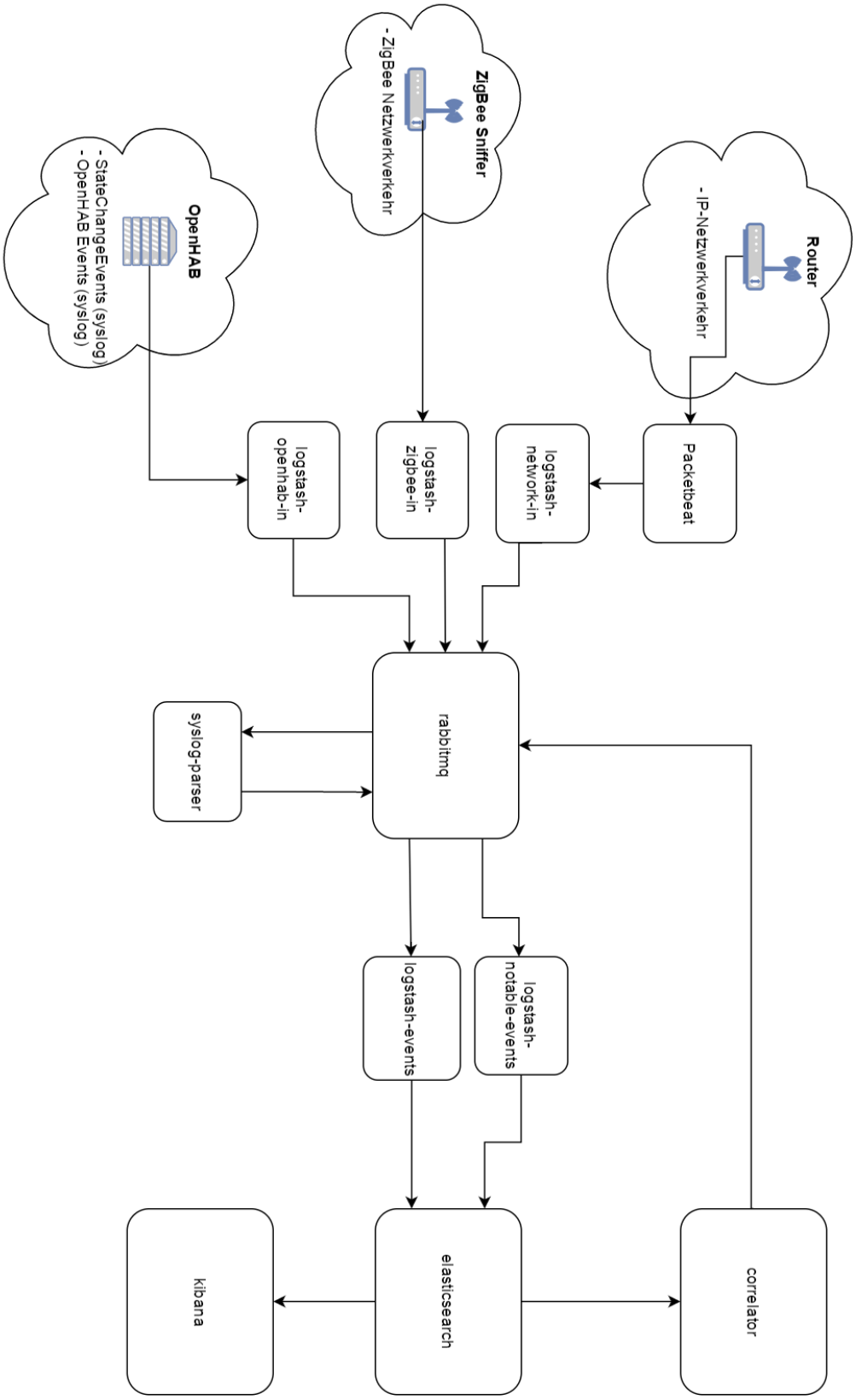


Abbildung 7.1: Aufbau des Smart Home SIEM-Systems (eigene Darstellung)

7.3 Daten sammeln - Proxy Dienste

Dieser Abschnitt beschreibt das Sammeln der in Tabelle 6.4 gezeigten Daten. Dabei werden sich ähnelnde und wiederholende Konfigurationen anhand des ersten Datentyps gezeigt und im weiteren Verlauf nur noch textuell beschrieben.

7.3.1 Netzwerkverkehr

Um den Netzwerkverkehr der gesamten auf Abbildung 5.1 gezeigten Versuchsumgebung zu sammeln, muss dieser zunächst an das SIEM-System, welches, wie in Tabelle 5.1 zu sehen, auf einem Host mit der IP-Adresse 192.168.1.189 läuft, geleitet werden. Dazu wird eine in der Arbeit von Tim Eckhardt beschriebene Methode angewandt [Eckhardt, 2017]. Auf dem mit OpenWRT ausgestatteten Router wird folgender Befehl ausgeführt, um ein- und ausgehenden Datenverkehr an den Host des SIEM-Systems weiterzuleiten.

```
iptables -t mangle -A PREROUTING -j TEE --gateway 192.168.1.189
iptables -t mangle -A POSTROUTING -j TEE --gateway 192.168.1.189
```

Da über diese Methode jedoch lediglich der Netzwerkverkehr weitergeleitet wird, der von außerhalb nach innerhalb des LAN oder umgekehrt läuft, muss iptables auch auf den anderen Hosts des Smart Home LAN ausgeführt werden.
Nachdem diese Konfiguration vorgenommen wurde, ermöglicht das dem Elastic Stack angehörige Tool Packetbeat, welches im gleichnamigen Docker-Container läuft, diese Daten zu sammeln. Für diese Arbeit werden aus dem Netzwerkverkehr, wie in Tabelle 6.4 zu sehen, HTTP- und Flow-Daten benötigt. Packetbeat wird dementsprechend konfiguriert, um die vom Router weitergeleiteten HTTP- und Flow-Daten zu sammeln. Im Anschluss kann Packetbeat die gesammelten Daten an das Tool Logstash weiterleiten, welches ebenfalls dem Elastic Stack angehört. Dazu wird in der Konfigurationsdatei packetbeat.yml folgende Einstellung vorgenommen:

```
output.logstash:
# The Logstash hosts
hosts: ["localhost:5044"]
```

Die logstash-network-in Komponente läuft auf Port 5044 des SIEM-System Hosts und nimmt die von Packetbeat gesammelten Daten darüber entgegen. Dazu ist folgende Konfiguration in der Datei logstash.conf notwendig:

```
input {
        beats {
        port => 5044
```

```
}}
```

Nachdem eine Bearbeitung der Daten durch Logstash vorgenommen wurde (siehe Abschnitt 7.5), werden diese durch das Tool im JSON-Format mit folgender Konfiguration an die rabbitmq Komponente weitergeleitet (logstash.conf):

```
output {
        rabbitmq {
                host => "rabbitmq"
                exchange => "traffic-in-exchange"
                exchange_type => "direct"
                key => "traffic"
                durable => true
                user => "XXXXX"
                password => "XXXXX"
}}
```

7.3.2 OpenHAB Status Events

Die OpenHAB Status Events treten in Form von Log-Daten im syslog-Format auf und werden von der OpenHAB Anwendung im Anwendungsverzeichnis als Datei abgelegt. Das macht es notwendig, diese Daten zunächst zum Host des SIEM-Systems zu senden, bevor sie dort verarbeitet werden können. Dazu wird auf dem OpenHAB Host (Raspberry Pi auf Abbildung 5) ein Syslog-ng[7] Server eingerichtet. Der Syslog-ng Server wird so konfiguriert, dass er die Daten aus den Log-Dateien der Anwendung per TCP auf Port 1025 des SIEM-System Hosts sendet (syslog-ng.conf):

```
source s_openHabEvents {
        file("/var/log/openhab2/events.log");
};
source s_openHabApplication {
        file("/var/log/openhab2/openhab.log");
};

destination d_siemSyslog {network("192.168.1.189"
transport("tcp") port("1025"));
};

log {source(s_openHabEvents); destination(d_siemSyslog); };
```

[7] Syslog-ng ist ein Open Source Log Management, welches mit dem syslog-Format arbeitet (https://syslog-ng.org/)

```
13 log {source(s_openHabApplication); destination(d_siemSyslog); };
```

Anschließend kann die logstash-openhab-in Komponente die Log-Daten auf Port 1025 entgegennehmen und der rabbitmq Komponente zur Verfügung stellen. Die Konfiguration dazu verläuft ähnlich der in Abschnitt 7.3.1 gezeigten.

7.3.3 ZigBee

Den ZigBee-Netzwerkverkehr verfügbar zu machen ist mit dem größten Aufwand verbunden. Da mit dem Tool Killerbee [Wright, 2009] zwar eine Möglichkeit zur Verfügung steht, die ZigBee-Daten zu speichern, diese aber anschließend in einem für die Weiterverarbeitung unbrauchbaren Format vorliegen, bedarf es des Einsatzes weiterer Tools. Das Tool Wireshark[8] ist in der Lage, den von Killerbee gespeicherten ZigBee-Verkehr zu interpretieren und im JSON-Format zu speichern, welches wiederum von Logstash verwendbar ist. Jedoch ist diese Methode nur manuell anstoßbar und nicht kontinuierlich automatisiert zu verwenden, sodass es vonnöten ist, ein eigenständiges Python-Skript zu entwickeln, welches beide Tools verwendet und im JSON-Format aufbereitete Daten produziert. Eine zusätzliche Schwierigkeit entsteht durch einen Firmware Bug des RZUSBSTICKs [riverloopsec, 2017].
Da der RZUSBSTICK bereits auf dem Host des SIEM-Systems läuft, ist keine Weiterleitung auf einen anderen Host notwendig. Das entstandene Skript legt die aufbereiteten Daten wiederum in einer Datei ab, welche durch Logstash interpretierbar ist und mit folgender Konfiguration an die rabbitmq Komponente geleitet wird (logstash.conf):

```
1  input {
2  file {
3  path => "/logs/tsharkfile"
4  start_position => "beginning"
5  codec => "json"
6  }}
7  output {
8  rabbitmq {
9  host => "rabbitmq"
10 exchange => "zigbee-in-exchange"
11 exchange_type => "direct"
12 key => "zigbee"
13 durable => true
14 user => "XXXXX"
15 password => "XXXXX"
```

[8] Wireshark ist ein weit verbreitetes Protokollanalyse Tool (https://www.wireshark.org/)

```
16 }}
```

7.4 Middleware

Die Middleware wird, wie bereits erwähnt, durch eine RabbitMQ abgebildet, deren Queues und Exchanges auf Abbildung 7.2 zu sehen sind. Eine Queue ist dabei als eine Art Warteschlange zu betrachten, die Events (gesammelte Log-Daten) solange hält, bis sie von der logstash-events, der logstash-notable-events oder der syslog-parser Komponente abgeholt werden. Dabei gilt das Prinzip First-in-first-out. Wird ein Event abgeholt, wird es aus der Queue entfernt. Exchanges dagegen sind als Schnittstellen der RabbitMQ zur Außenwelt zu sehen. So sprechen die im Abschnitt 7.3 beschriebenen Proxy Dienste und die correlator Komponente die RabbitMQ Exchanges an, welche die Events an die auf Abbildung 7.2 sichtbaren Queues weiterleiten.

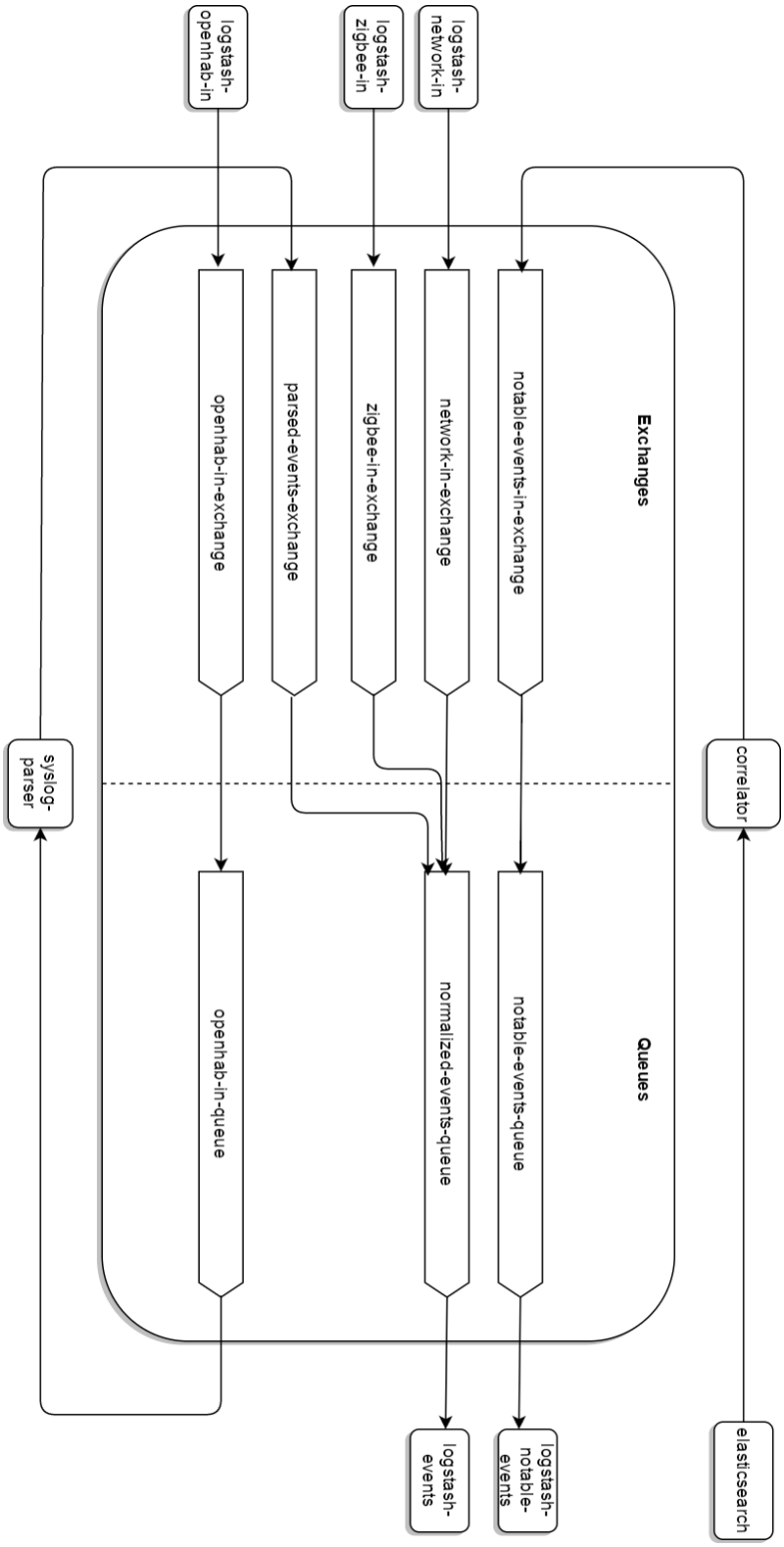


Abbildung 7.2: Architektur der Middleware (eigene Darstellung)

7.5 Normalisierung & Filterung

Die Normalisierung der Daten ist ein enorm wichtiger Schritt, kommen diese doch mit unterschiedlichsten Feldnamen und Formaten daher. In Kapitel 6.3.3 sind in Tabelle 6.4 die Felder der jeweiligen Datentypen zu erkennen, die für die Umsetzung der Use Cases benötigt werden und zwingend entsprechend der Vorgabe normalisiert werden müssen. Zusätzlich vorhandene Felder der Daten werden bis auf wenige, die eventuell nützliche Zusatzinformationen für die spätere Analyse beinhalten, gelöscht.
Das Filtern von kompletten Nachrichten wird ebenfalls in diesem Kapitel behandelt und soll Events, die nicht zur Erkennung der Use Cases dienen können, entfernen.
Die Normalisierung und das Filtern der Events geschieht technisch auf unterschiedlichen Wegen, die im Folgenden beschrieben sind.

7.5.1 Netzwerkverkehr

Logstash ist grundsätzlich in der Lage, JSON-Daten zu bearbeiten, indem Namen von Feldern und deren Inhalte geändert oder gelöscht werden. Auch kann Logstash Daten die definierten Kriterien entsprechen verwerfen und damit filtern. Bei der Bearbeitung des Netzwerkverkehrs kann durchgängig Logstash für die Normalisierung und das Filtern verwendet werden. Die einzelnen Normalisierungsschritte werden hier beispielhaft anhand der Flow-Daten gezeigt.

Flow-Daten

Die Flow-Daten werden wie bereits beschrieben durch Logstash normalisiert. Beispielhaft ist im Folgenden eine Konfiguration zu sehen, die bestimmte Felder der Flow-Daten, in diesem Fall IP-Adressen und Ports, gemäß der Vorgaben aus Kapitel 6.3.3 umbenennt (logstash.conf):

```
filter {
if [type] == "flow"{
mutate {
rename => { "ip" => "dest_ip" }
rename => { "client_ip" => "source_ip" }
rename => { "port" => "dest_port" }
rename => { "client_port" => "source_port" }
}
```

Auch das Filtern der Flow-Daten wird im Folgenden exemplarisch gezeigt und geschieht ebenfalls mithilfe von Logstash. Im hier gezeigten Beispiel wird die Kommunikation der Docker-

Container untereinander, welche sich in einem virtuellen Netzwerk auf dem SIEM-System Host befinden, gefiltert, da diese für das Erkennen der Use Cases nicht relevant sind (logstash.conf):

```
if [source][ip] =
~ /172.18.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?).
                (25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)/
        {
        drop {}}
```

Nach den Normalisierungs- und Filterschritten enthält ein Flow-Event die auf Abbildung 7.3 gezeigten Felder und Inhalte und erfüllt damit die Bedingungen aus Tabelle 6.4.

@timestamp		January 18th 2018, 14:46:30.000
@version		1
_id		AWEJhV4eRRpyf_JoMkOf
_index		testindex
_score		-
_type		flow
dest.stats.net_bytes_total		184
dest.stats.net_packets_total		2
dest_ip		91.189.94.4
dest_port		123
final		true
flow_id		EAL/////AP////8I//8AAAFbvV4EwKgBvXsAewA
last_time		January 18th 2018, 14:45:52.207
source.stats.net_bytes_total		276
source.stats.net_packets_total		3
source_ip		192.168.1.189
source_port		123
start_time		January 18th 2018, 14:45:52.190
transport		udp
type		flow

Abbildung 7.3: Normalisierte Flow-Daten (Kibana Screenshot, selbst erstellt)

HTTP

HTTP-Daten durchlaufen die Filter- und Normalisierungsschritte mit denselben Methoden wie die Flow-Daten und enthalten anschließend noch die auf Abbildung 7.4 gezeigten Felder und Inhalte. Damit sind auch hier die Bedingungen aus Tabelle 6.4 erfüllt.

@timestamp	January 18th 2018, 14:21:29.006
@version	1
_id	AWEJbnZpRRpyf_JoMfrA
_index	testindex
_score	-
_type	http
dest_ip	192.168.1.143
dest_port	80
method	PUT
path	/api/x9Z-zi5yhvUMwwAFxIKUq3jg6Vd1cuW-H5gpmSvL/lights/2/state
query	PUT /api/x9Z-zi5yhvUMwwAFxIKUq3jg6Vd1cuW-H5gpmSvL/lights/2/state
request	PUT /api/x9Z-zi5yhvUMwwAFxIKUq3jg6Vd1cuW-H5gpmSvL/lights/2/state HTTP/1.1 Host: 192.168.178.47 User-Agent: curl/7.55.1 Accept: */* Content-Type: application/json Content-Length: 21 {"on":true,"bri":150}
requested_host	192.168.178.47
response	HTTP/1.1 200 OK Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0 Pragma: no-cache Expires: Mon, 1 Aug 2011 09:00:00 GMT Connection: close Access-Control-Max-Age: 3600 Access-Control-Allow-Origin: * Access-Control-Allow-Credentials: true Access-Control-Allow-Methods: POST, GET, OPTIONS, PUT, DELETE, HEAD Access-Control-Allow-Headers: Content-Type Content-type: application/json [{"success":{"/lights/2/state/on":true}},{"success":{"/lights/2/state/bri":150}}]
response_code	200
responsetime	0
server	
source_ip	192.168.178.52
source_port	39,678
status	OK
type	http

Abbildung 7.4: Normalisierte HTTP-Daten (Kibana Screenshot, selbst erstellt)

7.5.2 OpenHAB Status Events

Die OpenHAB Status Events sind die einzigen in dieser Arbeit verwendeten Events, welche nicht per Logstash normalisiert werden können, sondern die syslog-parser Komponente verwenden. In der syslog-parser Komponente liest ein Python-Skript alle Events in der openhab-in-queue der RabbitMQ, normalisiert diese, und gibt sie in den parsed-events-exchange der RabbitMQ zurück. Aufgrund des benötigten Feldes 'eventtype', zu sehen in Tabelle 6.4 , ist es nicht möglich, die Normalisierung mit Logstash durchzuführen. Der Inhalt dieses Feldes muss aus dem String des originalen Log-Eintrags von OpenHAB geparsed werden, was mit Logstash nicht möglich ist. Nach dem Einsatz des Skriptes haben die in Elasticsearch gespeicherten OpenHAB Status Events das in Abbildung 7.5 zu sehende Format und erfüllen damit die Bedingungen.

@timestamp			January 17th 2018, 18:22:55.000
@version			1
_id			AWEFJg2urSvD7ctSVeuf
_index			testindex
_score			-
_type			openhab
eventtype			ItemStateChangedEvent
facility			1
facility_label			user-level
host			192.168.1.160
logsource			openHAB
message			2018-01-17 18:22:55.073 [ItemStateChangedEvent] - hue_0100_00178870c72d_2_brightness changed from 39 to 98
priority			13
real_timestamp			2018-01-17 18:22:55.073
severity			5
severity_label			Notice
timestamp			Jan 17 18:22:55
type			openhab

Abbildung 7.5: Normalisierte OpenHAB Status Events (Kibana Screenshot, selbst erstellt)

7.5.3 ZigBee

Die ZigBee-Daten können wie die Netzwerk-Daten mit Logstash gefiltert und normalisiert werden und haben nach diesem Prozess gemäß den Anforderungen aus Kapitel 6.3.3 die auf Abbildung 7.6 gezeigten Felder.

🕒	@timestamp	⊕ ⊖ ▥ ✱	January 18th 2018, 16:49:28.127
t	@version	⊕ ⊖ ▥ ✱	1
t	_id	⊕ ⊖ ▥ ✱	AWEJ9fCIgOuAwfnh1gYj
t	_index	⊕ ⊖ ▥ ✱	testindex
#	_score	⊕ ⊖ ▥ ✱	-
t	_type	⊕ ⊖ ▥ ✱	zigbee
t	dest_addr	⊕ ⊖ ▥ ✱	0x0000772c
t	dest_pan	⊕ ⊖ ▥ ✱	0x0000117d
t	real_timestamp	⊕ ⊖ ▥ ✱	Jan 18, 2018 16:15:33.769769000 CET
?	score	⊕ ⊖ ▥ ✱	⚠ -
t	source_addr	⊕ ⊖ ▥ ✱	0x0000a02d
t	type	⊕ ⊖ ▥ ✱	zigbee

Abbildung 7.6: Normalisierte ZigBee-Daten (Kibana Screenshot, selbst erstellt)

7.6 Speicherung

Die normalisierten und gefilterten Events sowie die Notable Events gelangen wie in Abschnitt 7.2.1 beschrieben in Elasticsearch. Die Strukturierung der Daten in Elasticsearch geschieht mithilfe sogenannter Indizes. Die logstash-events Komponente schreibt alle erhaltenen Events in den Index 'testindex', auf welchem später die Use Cases testweise Korrelationssuchen ausführen, und die logstash-notable-events Komponente in den 'notable-events' Index. Mithilfe dieser Trennung lassen sich Events und Notable Events im Zuge einer Analyse bzw. der Testdurchführung besser unterscheiden und betrachten.

7.7 Korrelation

Die Korrelation ist das Herzstück eines SIEM-Systems, ist sie doch für die Realisierung der Use Cases verantwortlich. In diesem SIEM-System werden die Use Cases in der Komponente correlator mithilfe von Python-Skripten umgesetzt, die im 'testindex' von Elasticsearch nach auffälligen Events suchen. Werden sie fündig, erstellen sie ein Notable Event, welches sie in den notable-events-exchange der RabbitMQ speichern (siehe Abbildung 7.2 und 7.1). Das Notable Event gelangt anschließend über die notable-events-queue in den notable-events Index von Elasticsearch. Die genaue Beschreibung der Korrelationssuchen, die für die Umsetzung der Use Cases entwickelt werden müssen, findet sich in Kapitel 8.

7.8 Analyse

Für die Analysephase, welche innerhalb dieser Arbeit nur zur Auswertung der Tests verwendet wird, wurde in den Anforderungen an das System eine Visualisierung gefordert. Diese wird im Smart Home SIEM-System mithilfe von Kibana umgesetzt, einem Tool, welches sich wie Logstash und Packetbeat dem Elastic Stack zuordnen lässt. Kibana bietet die Möglichkeit, Events und Notable Events über eine Webanwendung zu betrachten. Dabei sind bspw. die Abbildungen 7.3, 7.4, 7.5 und 7.6 aus Kibana entstanden. Auch bietet Kibana die Möglichkeit, sich grafische Auswertungen in Form von Dashboards zusammenzustellen, wie beispielhaft auf Abbildung 7.7 zu sehen. Dort sind auf der linken Seite die zwanzig am häufigsten angesprochenen IP-Adressen und rechts ein Verlauf von OpenHAB Status Events abgebildet.

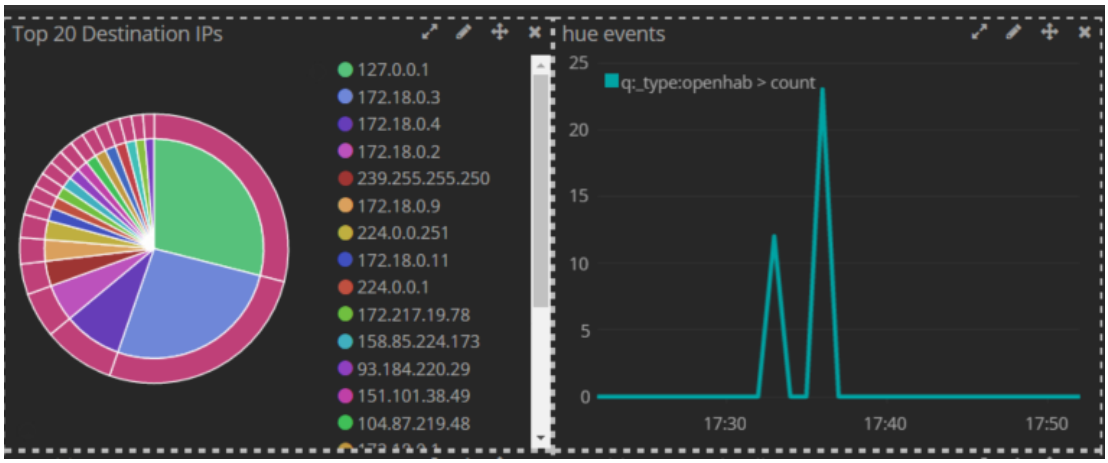


Abbildung 7.7: Kibana Dashboard (Kibana Screenshot, selbst erstellt)

8 Umsetzung der Use Cases

In diesem Kapitel wird die konkrete Implementierung der Use Cases (siehe Kapitel 6.3) beschrieben. Dazu wird zunächst auf die verwendeten Technologien und Methoden eingegangen, um anschließend die einzelnen Use Cases in ihrer Funktionsweise und Logik zu beschreiben. In Kapitel 9 wird getestet, ob die Use Cases im SIEM-System zur Erkennung der Angriffsstufen führen. Dabei werden nur diejenigen Use Cases umgesetzt und die dementsprechenden Angriffsstufen getestet, die in Kapitel 6.3 nicht aufgrund von Problemen aussortiert wurden.
Die Use Cases werden im in Kapitel 7 beschriebenen Docker-Container correlator beheimatet und ausgeführt. Zur Implementierung der einzelnen fachlichen Use Cases müssen eine oder mehrere Korrelationssuchen geschrieben werden. Diese suchen im Datenbestand des elasticsearch Containers nach Events, die auf bestimmte Kriterien zutreffen. Die einzelnen Ergebnisse dieser Korrelationssuchen müssen je nach Use Case im Anschluss kombiniert werden. Wie in [Lavrova und Pechenkin, 2015] beschrieben, gibt es unterschiedliche Typen von Korrelationssuchen. Im Rahmen dieser Arbeit werden regelbasierte und statistische Korrelationssuchen verwendet (Näheres in der Umsetzungserläuterung der einzelnen Use Cases).
Technisch werden die einzelnen Korrelationssuchen in der Elasticsearch Query DSL[1] geschrieben, einer Abfragesprache, welche anhand von im JSON-Format definierten Parametern den Datenbestand von Elasticsearch durchsucht.
Die definierten Korrelationssuchen müssen anschließend in Skripte verpackt werden, welche für die Kombination und Auswertung der Suchergebnisse, deren regelmäßige Ausführung und die Erzeugung der sogenannten Notable Events sorgen. Die Notable Events sollen genau dann entstehen und in der Benutzeroberfläche (Kibana) des SIEM-Systems angezeigt werden, wenn ein Use Case anschlägt. Für die technische Umsetzung dieser Skripte wird innerhalb dieser Arbeit die Programmiersprache Python eingesetzt. Erwähnenswerte, zusätzliche Bibliotheken, die innerhalb der Python-Skripte verwendet werden sind zum einen pika[2], welche für die Verbindung zur rabbitmq Komponente verantwortlich ist und zum anderen elasticsearch[3], welche in der Lage ist, die formulierten Korrelationssuchen im Elasticsearch Datenbestand

[1] https://www.elastic.co/guide/en/elasticsearch/reference/6.0/query-dsl.html
[2] https://pika.readthedocs.io/en/0.10.0/
[3] https://elasticsearch-py.readthedocs.io/en/master/

auszuführen und Ergebnisse zu erhalten. Abbildungen zu den einzelnen erzeugten Notable Events finden sich in Kapitel 9, wenn die Erkennung der Angriffsstufen getestet und in Folge dessen Notable Events erzeugt werden. Bei der Umsetzung wird jeder Use Case in ein eigenes Python-Skript verpackt.

8.1 Remote Control Attacke auf Hue Bridge & SSDP Reflection Attacke

Da in Kapitel 6 die Use Cases der beiden Angriffsszenarien Remote Control Attacke auf Hue Bridge und die SSDP Reflection Attacke aufgrund vieler gemeinsamer Kill Chain Stufen im selben Abschnitt behandelt wurden, werden sie auch bei der Umsetzung der Use Cases zusammengefasst. Anhand des ersten Use Cases wird exemplarisch näher auf die Syntax der Elasticsearch Query DSL eingegangen, welche für alle weiteren Use Cases ebenfalls Gültigkeit hat.

8.1.1 SSDP-Scan

Der erste zu implementierende Use Case dient dem Erkennen des SSDP-Scans. Dabei wurde in Kapitel 6.3 festgelegt, dass dieser Use Case aus einer Kombination zweier Korrelationssuchen entstehen soll. Die erste soll Nachrichten im Netzwerkverkehr entdecken, die von einem unerwarteten Gerät an Port 1900 der Broadcastadresse 239.255.255.250 gesendet wird, da dies ein Symptom eines SSDP-Scans darstellt.
Um unerwartete Geräte zu identifizieren, ist ein Blick auf die Geräteliste notwendig (Tabelle 5.1). Von der Philips Hue Bridge und dem Windows-Rechner werden SSDP-Scans erwartet, da es sich dabei um SSDP-fähige Geräte handelt. Somit handelt es sich im Umkehrschluss bei allen Geräten, die nicht diesen beiden Geräten entsprechen um unerwartete Geräte.
Der folgende JSON-Code entspricht der ersten Korrelationssuche.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "dest_ip": "239.255.255.250" }},
{"match": { "dest_port": "1900" }},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]
```

```
, "must_not": [
{"match": { "source_ip": "192.168.1.143"}},
{"match": { "source_ip": "192.168.1.229"}},
]}}}
```

Dabei wird zunächst der in Elasticsearch angelegte Index 'testindex' mit einem GET-Request angesprochen. Anschließend wird eine Query gebildet, die aus einem 'must'- und einen 'must_not'-Bereich besteht. Dabei sind sämtliche im 'must'-Bereich angegebenen Parameter für die gesuchten Ergebnisse erforderlich, wohingegen die im 'must_not'-Bereich aufgeführten Parameter nicht zutreffen dürfen. Im Fall des ersten Use Cases werden somit sämtliche Nachrichten, welche eine Ziel-Adresse (dest_ip) 239.255.255.250 mit Port 1900 (dest_port) haben, gefunden. Ausgeschlossen werden all jene Nachrichten, die von Quell-Adressen der beiden durch die Geräteliste ausgeschlossenen Geräte gesendet wurden. Unter 'range' wird angegeben, in welchem Zeitraum die Korrelationssuche nach passenden Events suchen soll. Im Fall der hier entwickelten Use Cases sind dies, wenn nicht anders beschrieben, die vergangenen 5 Minuten. Die Python-Skripte rufen die Abfragen der Use Cases ebenfalls alle 5 Minuten auf, was dazu führt, dass keine zeitlichen Lücken entstehen und keine Ereignisse mehrfach gefunden werden.

Die im Folgenden gezeigte zweite Korrelationssuche sucht nach Antworten der gescannten Geräte. Dabei werden als 'must_not'-Parameter die beiden ausgeschlossenen Geräte sowie die Broadcast-Adresse als Ziel-Adressen verwendet, da SSDP-Nachrichten an diese Geräte zum Normalverhalten gehören. Gleichzeitig werden diese Geräte als benötigte Quell-Adressen angegeben, da nach Nachrichten, die von SSDP-fähigen Geräten an unerwartete Geräte gesendet werden, gesucht werden soll. Mit dem 'should'-Parameter, der den 'must'-Parameter ersetzt, und der 'minimum_should_match'-Variablen 2 wird dabei angegeben, dass zwei der drei angegebenen 'match'-Ausdrücke zutreffen müssen. Das ist an dieser Stelle notwendig, da eine gefundene Nachricht nicht zwei unterschiedliche 'dest_ip's haben kann.

```
GET testindex/_search
{
"query": {
"bool": {
"should": [
{"match": { "dest_port": "1900" }},
{"match": { "source_ip": "192.168.1.143"}},
{"match": { "source_ip": "192.168.1.229"}}],
"minimum_should_match": 2,
"must": [
```

```
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]
, "must_not": [
{"match": { "dest_ip": "239.255.255.250"}},
{"match": { "dest_ip": "192.168.1.143"}},
{"match": { "dest_ip": "192.168.1.229"}},
]}}}
```

Bei den beiden hier eingesetzten Korrelationssuchen handelt es sich um regelbasierte Korrelationen, da nach konkreten Ereignissen bzw. Ereigniskombinationen gesucht wird, deren Eigenarten bereits zuvor bekannt sind. Einer der Vorteile dieser Art der Korrelationssuche ist die mögliche präzise Suche nach Ereignissen. Zum Nachteil kann insbesondere im operativen Betrieb eines SIEM-Systems die Tatsache werden, dass die Korrelationen regelmäßig gewartet, erweitert oder neu geschrieben werden müssen [Lavrova und Pechenkin, 2015]. Da im Fall dieses Use Cases die Parameter der Angriffsstufen bereits herausgearbeitet wurden, eignet sich diese Methode zur Umsetzung.
Wenn beide Korrelationssuchen Treffer aufweisen, könnte ein SSDP-Scan von einem unerwarteten Gerät erfolgt sein, der Use Case schlägt an und erzeugt ein Notable Event.
Voraussetzung, um nicht in False Positives zu ertrinken, ist es, bei der Entwicklung dieses Use Cases alle SSDP-fähigen Geräte auszuschließen, die legalerweise SSDP-Nachrichten verschicken. Werden diese Geräte ausgeschlossen, entsteht jedoch eine Grauzone und SSDP-Angriffe werden, von den ausgeschlossenen Geräten ausgeführt, nicht erkannt.

8.1.2 Abfluss von Informationen

Der Use Case zum Abfluss von Informationen ist, wie in Kapitel 6.3 bereits erwähnt, ein sehr unsicherer Use Case, da mit den innerhalb dieser Arbeit verwendeten Mitteln lediglich nach HTTP-Kommunikation mit Servern gesucht werden kann, die bereits als verdächtig bekannt und in bspw. Threat Intelligence Feeds veröffentlicht wurden. Wird eine solche Blacklist in die ebenfalls regelbasierten Korrelationssuchen eingepflegt, schlagen die Use Cases bei jeder Kommunikation mit einem dieser Server an. In diesem Fall wurde für diesen sehr simplen Use Case eine exemplarische Adresse, die sich außerhalb des Smart Home LAN befindet, ausgewählt (217.13.68.251, die Adresse eines großen Online Nachrichtenportals). Wird diese Adresse im HTTP-Netzwerkverkehr gefunden, schlägt der Use Case an und ein Notable Event wird erzeugt.

```
GET testindex/_search
{
```

```
"query": {
"bool": {
"must": [
{"match": { "type": "http"}},
{"match": { "dest_ip": "217.13.68.251" }},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]}}}
```

8.1.3 Port Mapping

Der Use Case zur Erkennung eines möglichen Port Mappings ist ähnlich aufgebaut wie der zum SSDP-Scan zugehörige. Dabei wird entsprechend der Anforderung aus Kapitel 6.3 nach Nachrichten gesucht, die den Port 1900 des Routers ansprechen. Die Adresse des Routers, 192.168.1.1, kann aus der Geräteliste entnommen werden, genau wie die beiden erwarteten SSDP-Geräte, welche wieder ausgeschlossen werden. Natürlich wird bei dem hier abgedeckten Fall auch der SSDP-Scan Use Case anschlagen. Jedoch ist es wichtig, eine zusätzliche Nachricht zu bekommen, wenn der Router angesprochen wird, da er aufgrund der Tatsache, dass er den Schutz zur Außenwelt darstellt (siehe Kapitel 3.2) und der Möglichkeit des Port Mappings eine besondere Kritikalität hat.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "dest_ip": "192.168.1.1" }},
{"match": { "dest_port": "1900" }},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]
, "must_not": [
{"match": { "source_ip": "192.168.1.143"}},
{"match": { "source_ip": "192.168.1.229"}},
]}}}
```

Bei der Entwicklung dieses Use Cases besteht die selbe Problematik der Gratwanderung zwischen False Positives und akzeptierten Grauzonen wie beim SSDP-Scan Use Case.

8.1.4 Fehlgeschlagenes Bedienen der Philips Hue Bridge

Eine höhere Komplexität weisen die beiden Use Cases zum erfolglosen und erfolgreichen Bedienen der Philips Hue Bridge auf. Für den ersten, welcher lediglich den Bedienungsversuch der Philips Hue Bridge abbildet, müssen vor der Umsetzung der Korrelationssuche ein paar Voraussetzungen geschaffen werden. Zunächst ist es wichtig, eine beispielhafte HTTP-Nachricht zur Verfügung zu haben, bei der die Authentifizierung an der Philips Hue Bridge API erfolglos war. Dazu wurde innerhalb dieser Arbeit ein Skript geschrieben, welches in Kapitel 9 genauer beschrieben wird und HTTP-Nachrichten generiert, die gegen die Philips Hue Bridge API geschickt werden können. Wird im HTTP-Request ein falscher Benutzername gesetzt (orange markiert), wird dieser erfolglose Bedienungsversuch, wie auf Abbildung 8.1 zu sehen, an Elasticsearch geschickt und in Kibana angezeigt. Für die Entwicklung dieses Use Cases ist insbesondere der Inhalt der Antwort (’response’ auf der Abbildung) von Bedeutung.

@timestamp		January 7th 2018, 16:23:22.716
@version		1
_id		AWDROBpn5ZhDUmusJGN1
_index		testindex
_score		-
_type		http
dest_ip		192.168.1.143
dest_port		80
method		PUT
path		/api/aksdhawjd1212/lights
query		PUT /api/aksdhawjd1212/lights
request		PUT /api/aksdhawjd1212/lights HTTP/1.1 Host: 192.168.1.143 User-Agent: curl/7.55.1 Accept: */*
requested_host		192.168.1.143
response		HTTP/1.1 200 OK Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0 Pragma: no-cache Expires: Mon, 1 Aug 2011 09:00:00 GMT Connection: close Access-Control-Max-Age: 3600 Access-Control-Allow-Origin: * Access-Control-Allow-Credentials: true Access-Control-Allow-Methods: POST, GET, OPTIONS, PUT, DELETE, HEAD Access-Control-Allow-Headers: Content-Type Content-type: application/json [{"error":{"type":1,"address":"/lights","description":"unauthorized user"}}]

Abbildung 8.1: Philips Hue Bridge: Failed Access (Kibana Screenshot, selbst erstellt)

Im folgenden JSON-Code ist die Umsetzung der Korrelationssuche zu sehen:

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "type": "http"}},
{"match": { "dest_ip": "192.168.1.143"}},
{"match": { "requested_host": "192.168.178.47"}},
{"match_phrase": { "response": "unauthorized user"}},
```

```
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]
, "must_not": [
{"match": { "requested_host": "192.168.1.143"}}
]}}}
```

Dabei wird die Suche zunächst auf den Typ HTTP eingegrenzt, da nur dieser von Relevanz sein kann. Die Ziel-Adresse ist die IP-Adresse der Philips Hue Bridge, wie in Tabelle 5.1 zu sehen. Der Parameter 'requested_host' wird im 'must'-Bereich mit der WAN- und im 'must_not'-Bereich mit der direkten Adresse der Philips Hue Bridge belegt, um nur Nachrichten zu finden, die von außerhalb des Smart Home LAN über die WAN-Adresse des Routers gesendet wurden. Entscheidend ist nun der Eintrag im Feld 'match_phrase'. Hier wird ein eindeutiger Teilstring der auf Abbildung 8.1 abgebildeten 'response' eingetragen, nach welchem gesucht werden soll. So wird sichergestellt, dass der Use Case lediglich bei fehlgeschlagenen Bedienversuchen anschlägt. Diese könnten in hoher Anzahl auch auf Brute Force Angriffe hinweisen, welche das Ziel haben, den korrekten Benutzernamen für die Philips Hue Bridge API herauszufinden. Auch bei dieser Korrelationssuche handelt es sich um eine regelbasierte Korrelation. Das erzeugte Notable Event wird zusätzlich mit der Anzahl der entdeckten Zugriffsversuche ausgestattet (siehe Abbildung 9.9), welche durch statistische Methoden gesammelt werden. Eine mögliche Erweiterung mithilfe einer statistischen Korrelation könnte anhand von festgelegten Grenzwerten und der gesammelten Anzahl von Zugriffsversuchen ermitteln, ob es sich bei den Versuchen um Brute Force Angriffe handelt.

8.1.5 Erfolgreiches Bedienen der Philips Hue Bridge

Der Use Case zur Entdeckung einer erfolgreichen Bedienung der Philips Hue Bridge von außerhalb des Smart Home LAN setzt, wie der vorangegangene Use Case, weitergehende Informationen für die Umsetzung voraus. Zunächst muss eine HTTP-Nachricht, die zu einem erfolgreichen Bedienen der Philips Hue Bridge geführt hat, ausgewertet werden. Auf Abbildung 7.4 ist eine solche zu sehen, die durch den normalen Bedienweg über OpenHAB ausgelöst wurde. Dabei wurde die Helligkeit der Glühbirne verändert. Auch hier ist insbesondere der Inhalt des 'response' Feldes von Interesse.
Laut der Anforderung aus Kapitel 6.3 besteht dieser Use Case aus der Kombination zweier Korrelationssuchen. Die erste ist analog zum vorangegangenen Use Case, dem fehlgeschlagenen Bedienversuch der Philips Hue Bridge, zu sehen. Dabei wird an dieser Stelle lediglich der Inhalt der 'match_phrase' auf 'success' geändert, damit nur diejenigen Nachrichten gefunden werden,

welche die API erfolgreich angesprochen haben. Die restlichen Parameter sind mit dem selben Inhalt gefüllt:

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "type": "http"}},
{"match": { "dest_ip": "192.168.1.143"}},
{"match": { "requested_host": "192.168.178.47"}},
{"match_phrase": { "response": "success"}}
]
, "must_not": [
{"match": { "requested_host": "192.168.1.143"}}
]}}}
```

Die zweite Korrelationssuche bezieht zum ersten Mal die von OpenHAB produzierten Status Events ein. Um sicherzugehen, dass der erfolgreiche Bedienversuch der Philips Hue API von außerhalb des LAN auch zu einer physischen Änderung des Zustandes der Philips Hue Glühbirne geführt hat, können diese Status Events herangezogen werden. Auch hier muss zunächst ein Status Event betrachtet werden, das während der Bedienung über den normalen Weg, also über OpenHAB, entstanden ist und in Elasticsearch gespeichert wurde (siehe Abbildung 7.5). Dabei ist insbesondere der Inhalt des normalisierten Feldes 'eventtype' von Bedeutung. Wird der Status einer Glühbirne geändert, wird ein 'ItemStateChangedEvent' erzeugt - unabhängig davon, ob die Änderung durch OpenHAB oder andere Anwendungen bzw. einen Angreifer zustande kam.

⏲	@timestamp	January 7th 2018, 17:34:55.000
t	@version	1
t	_id	AWDReZl-5ZhDUmusJcfV
t	_index	testindex
#	_score	-
t	_type	openhab
t	eventtype	ItemStateChangedEvent
#	facility	1
t	facility_label	user-level
t	host	192.168.1.160
t	logsource	openHAB
t	message	2018-01-07 17:34:55.053 [ItemStateChangedEvent] - hue_0100_00178870c72d_2_brightness changed from 67 to 79
#	priority	13
t	real_timestamp	2018-01-07 17:34:55.053
#	severity	5
t	severity_label	Notice
t	timestamp	Jan 7 17:34:55
t	type	openhab

Abbildung 8.2: OpenHAB State Change Event (Kibana Screenshot, selbst erstellt)

Entsprechend dieser Erkenntnisse wird die zweite Korrelationssuche wie im Folgenden zu sehen aufgebaut:

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "type": "openhab"}},
{"match": { "eventtype": "ItemStateChangedEvent"}},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]}}}
```

Dabei wird zunächst der Typ auf OpenHAB Status Events eingeschränkt. Anschließend wird nach 'ItemStateChangedEvents' gesucht. Wird ein solches Event gefunden, wurde neben der erfolgreichen HTTP-Nachricht auch ein OpenHAB Status Event erzeugt, dass die Änderung des physischen Zustandes der Glühbirne dokumentiert.
Beide hier verwendeten Korrelationssuchen sind ebenfalls regelbasierte Korrelationen.

8.1.6 SSDP Reflection Attacke

Die SSDP Reflection Attacke wird, wie in den Anforderungen beschrieben, mithilfe zweier Korrelationssuchen abgebildet. Die beiden Korrelationssuchen stehen in diesem Fall aber jeweils für sich alleine und werden im Python-Code nicht mit einer 'Und-Verknüpfung' verbunden. Die erste dieser beiden Korrelationssuchen soll eine ungewöhnlich hohe Menge an SSDP-Nachrichten, also Nachrichten an Port 1900 eines beliebigen Geräts, erkennen. Um eine solche Anomalie zu erkennen, wird eine statistische Korrelationsmethode verwendet [Lavrova und Pechenkin, 2015]. Ob sich das Zielgerät inner- oder außerhalb des LAN befindet spielt hier zunächst keine Rolle, da beide Varianten die Symptome einer SSDP Reflection Attacke darstellen könnten (siehe Kapitel 6.3).
Um dieses Ziel zu erreichen, werden zwei Queries eingesetzt. Die erste, im Folgenden zu sehen, summiert alle Nachrichten an Port 1900 eines beliebigen Gerätes im Zeitraum zwischen den vergangenen 10 Minuten bis zu den vergangenen 5 Minuten auf. Dabei werden die letzten 5 Minuten zum aktuellen Zeitpunkt ausgespart.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "dest_port": "1900" }},
{"range": {"@timestamp": {"gte": "now-10m","lte": "now-5m"}}}
]}}}
```

Mit der Summe der innerhalb dieses Zeitraums entstandenen SSDP-Nachrichten wird das Ergebnis der zweiten Query in Beziehung gesetzt. Dabei sucht die zweite Query über die vergangenen 5 Minuten. Im Python-Code wird an dieser Stelle ein Vergleich zwischen beiden Summen angestellt. Entspricht das Ergebnis aus Query 2 mindestens dem fünffachen des Ergebnisses der Query 1, schlägt der Use Case an und ein Notable Event wird erzeugt. Damit wäre gezeigt, dass sich der SSDP-Verkehr innerhalb der letzten 10 Minuten mehr als verfünffacht hat - was eine Anomalie darstellt.

Dieser Use Case hat mit ähnlichen Problemen wie alle SSDP-Angriffsstufen zu kämpfen. Der Erhöhungsfaktor der SSDP-Nachrichten, anhand welchem die Angriffsstufe erkannt werden soll, muss nach gesammelten Erfahrungswerten eingestellt werden, um weder Angriffe zu übersehen, noch ein eigentlich normales Verhalten im Netzwerk als Anomalie zu werten.

```
GET testindex/_search
```

```
{
"query": {
"bool": {
"must": [
{"match": { "dest_port": "1900" }},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]}}}
```

Die andere Möglichkeit der Erkennung dieser Angriffsstufe besteht in der Suche nach Nachrichten, die den Port 1900 eines Gerätes außerhalb des LAN ansprechen, da SSDP-Nachrichten im Normalverhalten lediglich innerhalb des LAN agieren (siehe Kapitel 6). Dazu werden im folgenden JSON-Code bereits aus vorangegangenen Use Cases bekannte Parameter eingesetzt. Zusätzlich wird im 'must_not'-Bereich der 'match_phrase'-Parameter so gesetzt, dass lediglich Ziel-Adressen außerhalb des Smart Home LAN gefunden werden. Auch die Broadcast-Adresse wird erneut ausgeschlossen.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "dest_port": "1900"}},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]
, "must_not": [
{"match_phrase": { "dest_ip": "192.168.1.1"}},
{"match": { "dest_ip": "239.255.255.250"}}
]}}}
```

8.2 ZigBee Take Over Attacke

Der zweite Block der Use Cases betrachtet die in Kapitel 6.3 herausgearbeiteten Stufen des ZigBee Take Over Angriffs (siehe auch Tabelle 6.3).

8.2.1 Factory Reset der Glühbirnen

Der erste Use Case zum Factory Reset der Glühbirnen wird laut der in Kapitel 6.3 gestellten Anforderung mit Hilfe dreier miteinander kombinierter Korrelationssuchen abgebildet.

Grundsätzlich geht es darum zu erkennen, wenn die Philips Hue Glühbirne vom eigenen PAN entfernt wird. Dazu sollen die ersten beiden Korrelationssuchen durch OpenHAB Status Events erkennen, wenn die Glühbirne für OpenHAB nicht mehr erreichbar ist. Hier müssen zunächst OpenHAB Status Events betrachtet werden, die diesen Fall darstellen. Auf Abbildung 8.3 ist ein solches Status Event zu sehen, welches durch simples Ausschalten der Glühbirne erzeugt wurde. Dabei sind insbesondere die Felder 'eventtype' und 'message' interessant.

@timestamp	January 18th 2018, 22:22:45.000
t	@version	1
t	_id	AWENivxtqXBME17u2eky
t	_index	testindex
#	_score	-
t	_type	openhab
?	eventtype	hingStatusInfoChangedEvent
?	facility	1
?	facility_label	user-level
?	host	192.168.1.160
?	logsource	openHAB
?	message	2018-01-18 22:22:45.050 [hingStatusInfoChangedEvent] - 'hue:0100:00178870c72d:2' changed from ONLINE to OFFLINE: Bridge reports light as not reachable
?	priority	13
t	real_timestamp	2018-01-18 22:22:45.050
?	severity	5
?	severity_label	Notice
?	timestamp	Jan 18 22:22:45
t	type	openhab

Abbildung 8.3: OpenHAB Bulb not reachable (eigene Darstellung)

Im folgenden JSON-Code wird der entsprechende 'eventtype' und ein eindeutiger Teil der 'message' als Suchparameter übernommen.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "type": "openhab"}},
{"match": { "eventtype": "hingStatusInfoChangedEvent"}},
{"match_phrase": { "message": "ONLINE to OFFLINE"}},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]}}}
```

Da dieses Status Event, wie zuvor beschrieben, auch durch ein simples Ausschalten der Glühbirne erzeugt wird und damit einem legalen Vorgang unterliegt, bedarf es einer Kombination mit einer zweiten Korrelationssuche, um den Use Case auszulösen. Dabei soll geprüft werden, ob die Glühbirne trotz der Beendigung der Verbindung zu OpenHAB weiterhin ZigBee-Nachrichten versendet. Das wäre nicht der Fall, wenn die Glühbirne lediglich ausgeschaltet wäre. Um das zu erreichen, wird in der folgenden Korrelationssuche zunächst der Typ auf 'zigbee' eingeschränkt. Aus der Geräteliste (siehe Tabelle 5.1) ist die ZigBee-Adresse der Glühbirne zu entnehmen. Diese wird in der Suche als Quell-Adresse eingetragen.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "type": "zigbee"}},
{"match": { "source_addr": "0x0000a02d"}},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]}}}
```

Nun könnte es sein, dass die Glühbirne zwar von OpenHAB getrennt, aber anschließend wieder verbunden wurde und darum ZigBee-Nachrichten innerhalb des Suchzeitraums versendet hat. Um diesen Fall auszuschließen, wird eine dritte Korrelationssuche angefertigt, welche nach einem anderen OpenHAB-Nachrichtentyp sucht: der Verbindung einer Glühbirne zu OpenHAB, in deren 'message' der String 'not reachable to Online' enthalten ist.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
```

```
{"match": { "type": "openhab"}},
{"match": { "eventtype": "hingStatusInfoChangedEvent"}},
{"match_phrase": { "message": "not reachable to ONLINE"}},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
]}}}
```

Die drei Korrelationssuchen werden im Python-Code verbunden, indem die erste und die zweite erfüllt sein müssen, die dritte jedoch ohne Treffer bleiben muss, um den Use Case auszulösen. Ist dies der Fall, wird ein Notable Event erzeugt.
Bei den hier verwendeten Korrelationssuchen handelt es sich um regelbasierte Korrelationen.

8.2.2 Glühbirne wurde zu fremdem PAN verbunden

Der letzte umzusetzende Use Case innerhalb dieser Arbeit dient der Entdeckung der Verbindung einer Philips Hue Glühbirne zu einem fremden PAN. Dazu werden zwei Korrelationssuchen eingesetzt.
Bei der ersten Korrelationssuche, die durch den folgenden JSON-Code abgebildet wurde, wird zunächst nach allen ZigBee-Nachrichten gesucht, welche die Glühbirne als Quelle haben (source_addr):

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "type": "zigbee"}},
{"match": { "source_addr": "0x0000a02d"}},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
],
"must_not": [
{"match": { "dest_pan": "0x0000117d"}}
]}}}
```

Im 'must_not'-Bereich der Suche wird als Ziel-PAN das eigene PAN (0x0000117d, siehe Tabelle 5.1) eingetragen. Das bedeutet, dass nach allen ZigBee-Nachrichten gesucht wird, die als Quelle zwar die Philips Hue Glühbirne haben, jedoch nicht das eigene PAN. Das wäre bspw. dann der Fall, wenn sich die Glühbirne, wie in Kapitel 6 beschrieben, versucht zu einem fremden PAN zu verbinden oder schon zum einem solchen verbunden ist.

Als zusätzliche Korrelationssuche wird der umgekehrte Fall gewählt, nämlich dass die Glühbirne das Ziel einer ZigBee-Nachricht ist, die wie im zuvor beschriebenen Fall ein fremdes

PAN als Ziel-PAN definiert hat. Eine solche Nachricht würde bedeuten, dass sich die Philips Hue Glühbirne bereits im fremden PAN befindet und deshalb über ein fremdes PAN adressiert wird.

```
GET testindex/_search
{
"query": {
"bool": {
"must": [
{"match": { "type": "zigbee"}},
{"match": { "dest_addr": "0x0000a02d"}},
{"range": {"@timestamp": {"gte": "now-5m","lte": "now"}}}
],
"must_not": [
{"match": { "dest_pan": "0x0000117d"}}
]}}}
```

Beide Korrelationssuchen deuten auf eine Übernahme der Philips Hue Glühbirne hin, sind regelbasierte Korrelationen und erzeugen bei Auslösung ein Notable Event.

9 Testen der Angriffserkennung

Nachdem in Kapitel 8 die Implementierung mit der technischen Umsetzung der Use Cases abgeschlossen wurde, soll in diesem Kapitel die Fähigkeit des SIEM-Systems, Angriffe zu entdecken gezeigt werden.
Konkretes Ziel der SIEM Übertragung ins Smart Home war, drei ausgewählte Angriffsszenarien (siehe Kapitel 6), deren Angriffsstufen sich insgesamt über alle drei IoT-Layer erstrecken, zu erkennen. Da die Angriffsszenarien vor der Konzeption und Planung der Use Cases in Kill Chains aufgebrochen wurden (siehe Abbildung 6.1), werden die Kill Chain Stufen als Basis für die Durchführung der Tests verwendet. In Kapitel 6 wurde gezeigt, dass sich die ersten drei Kill Chain Stufen 'Reconnaissance: Gather Information of the target', 'Weaponize the information' und 'Deliver the weapon' nicht vom SIEM Ansatz erkennen lassen. Tests zu diesen Stufen sind demnach auch kein Bestandteil dieses Kapitels. Die Remote Control Attacke auf Hue Bridge und die SSDP Reflection Attacke werden in den ersten beiden zu testenden Kill Chain Stufen zusammen betrachtet, da sie an diesen Stellen die gleichen Angriffsstufen beinhalten. Die 'ZigBee Take Over Attacke' hat nur in der ersten hier betrachteten Kill Chain Stufe Relevanz, da in der zweiten keine Angriffsstufe vorliegt und die finale nicht vom SIEM-System erkannt werden kann (siehe Kapitel 6.3).
Die Tests werden in der auf Abbildung 5.1 gezeigten Versuchsumgebung durchgeführt. Getestet wird mit unterschiedlichen Methoden, die sich an den in Kapitel 8 definierten Testmethoden orientieren (siehe auch Tabellen 6.1, 6.2 und 6.3). Dabei geht es grundsätzlich um ein Nachstellen der einzelnen Angriffsstufen, um typische Symptome zu erzeugen, die von den Use Cases erkannt werden können. Wird nach dem Durchführen eines Tests ein Notable Event vom SIEM-System erzeugt, hat ein Use Case aufgrund der aufgetretenen Symptome gegriffen und die nachgestellte Angriffsstufe erkannt.

9.1 Install, spread and hide

9.1.1 Remote Control Attacke auf Hue Bridge & SSDP Reflection Attacke

SSDP-Scan Auf der Kill Chain Stufe 'Install, spread and hide' wird in diesen beiden Angriffsszenarien zunächst die Malware installiert, was vom SIEM-Ansatz nicht zu entdecken ist, da diese noch nicht bekannt ist. Die nächste Stufe, der vom infizierten Gerät ausgehende SSDP-Scan, wird im Folgenden getestet.
Dazu wird ein Kali Linux[1] Rechner (IP: 192.168.1.243) in das LAN der Versuchsumgebung gehängt. Mithilfe des Kali Linux Tools Miranda wird, wie auf Abbildung 9.1 zu sehen, ein SSDP Scan gestartet, woraufhin sich die Philips Hue Bridge zurückmeldet.

Abbildung 9.1: Manuell ausgeführter SSDP-Scan (Kali Linux Screenshot, selbst erstellt)

Von dem Kali Linux Rechner wird, wie von einem infizierten Smartphone im originalen Angriffsszenario, kein SSDP-Scan erwartet. Als Reaktion auf den Scan erzeugt das Smart Home SIEM-System ein Notable Event, welches auf Abbildung 9.2 zu sehen ist. Darin sind zum einen 'EventIDs' zu sehen, welche auf die Ereignisse schließen lassen, die zum Auslösen des Use Cases geführt haben. Zum anderen sind neben einem 'NotableEventtype' und einer Beschreibung des Vorfalls im Feld 'message' die IPs aufgeführt, von welchen der Scan ausging ('effectedSourceIPs') und von welchen eine Antwort auf den Scan gesendet wurde ('EffectedResponseIPs').
Mit dem Erzeugen des korrekten Notable Events im Testfall wurde diese Angriffsstufe erkannt.

[1] Kali Linux ist eine Linux Distribution, die zum Penetration Testing entwickelt wurde und eine Vielzahl dazu geeigneter Tools vereint (https://www.kali.org/)

@timestamp	January 17th 2018, 18:31:08.140
@version	1
_id	AWEFLKgErSvD7ctSVgYv
_index	notable-events-index
_score	-
_type	notableEvent
effectedEventIDs	_id: AWEFKBxErSvD7ctSVfQc | _id: AWEFKENSrSvD7ctSVfRs | _id: AWEFKJGfrSvD7ctSVfVP | _id: AWEFKQa1rSvD7ctSVfc0 | _id: AWEFK1CQrSvD7ctSVgJ- | _id: AWEFKBxErSvD7ctSVfQG | _id: AWEFKXwDrSvD7ctSVfkN | _id: AWEFKco9rSvD7ctSVfqX | _id: AWEFK8XGrSvD7ctSVgPg | _id: AWEFLIkXrSvD7ctSVgXh
effectedEventIDs1	⚠ _id: AWEFKENSrSvD7ctSVfRt | _id: AWEFKJGfrSvD7ctSVfVQ | _id: AWEFKBxErSvD7ctSVfQd | _id: AWEFKaLtrSvD7ctSVfnp | _id: AWEFKXwDrSvD7ctSVfkP | _id: AWEFKS2-rSvD7ctSVffI | _id: AWEFKN-hrSvD7ctSVfY4 | _id: AWEFKLiJrSvD7ctSVfXx | _id: AWEFKGpXrSvD7ctSVfTK | _id: AWEFKQa1rSvD7ctSVfc1
effectedResponseIPs	response_ip: 192.168.1.143 | response_ip: 192.168.1.143 | response_ip: 192.168.1.143 | response_ip: 192.168.1.143 | response_ip: 192.168.1.143 | response_ip: 192.168.1.143 | response_ip: 192.168.1.143 | response_ip: 192.168.1.143 | response_ip: 192.168.1.143
effectedSourceIPs	source_ip: 192.168.1.243 | source_ip: 192.168.1.243 | source_ip: 192.168.1.243 | source_ip: 192.168.1.243 | source_ip: 192.168.178.51 | source_ip: 192.168.178.51 | source_ip: 192.168.178.51 | source_ip: 192.168.1.243 | source_ip: 192.168.178.51 | source_ip: 192.168.178.51
message	One or multiple hosts executed suspicious ssdp scans
notableEventtype	Suspicious SSDP Scan detected
type	notableEvent

Abbildung 9.2: Notable Event: SSDP-Scan (Kibana Screenshot, selbst erstellt)

Netzwerkverkehr mitschneiden Die Stufe 'Netzwerkverkehr mitschneiden' der 'Remote Control Attacke auf Hue Bridge' wurde ausgeschlossen, da eine Prozessüberwachung von Endgeräten nicht Bestandteil dieses Ansatzes ist und damit nicht getestet werden muss.

9.1.2 ZigBee Take Over Attacke

Bei der Vorbereitung der Tests zur ZigBee Take Over Attacke muss festgestellt werden, dass die ursprünglich in Kapitel 6.3 geplante Methode des Factory Resets der Glühbirne wie bspw. in [Community, 2017] gezeigt, nach Weiterentwicklungen des Herstellers nicht mehr ohne zusätzliche Hardware möglich ist. Da die wirkliche Nachstellung des gesamten Angriffsszenarios oder die Erzeugung von ZigBee-Nachrichten, um diese in den ZigBee-Netzwerkverkehr einzuschleusen, mit weiterem großen Aufwand verbunden wäre und der Schwerpunkt dieser Arbeit nicht verlassen werden soll, wurde an dieser Stelle die Entscheidung getroffen, ZigBee-Nachrichten so zu manipulieren, dass sie die Symptome der ZigBee Take Over Attacke zeigen. Die Manipulation findet dabei auf dem Host des SIEM-Systems, vor dem Einlesen und Verarbeiten der Nachrichten durch die logstash-zigbee-in Komponente statt. Mit dieser Methode kann das Angriffsszenario zwar nicht mit 'echten' ZigBee-Nachrichten getestet werden - auf die Funktionalität des SIEM-Systems bezogen steht der Test einer solchen Variante aber in nichts nach, da die Manipulation vor dem ersten Eingreifen des SIEM-Systems vorgenommen wird. Für zukünftige Arbeiten wäre es dennoch eine wertvolle Aufgabe, ZigBee-Angriffsszenarien vollständig nachzubilden und daraus Penetrationtests zu entwickeln.

Factory Reset der Glühbirne Auf dieser Stufe der Kill Chain findet zunächst der Factory Reset der Glühbirne statt (Abbildung 6.1). Um die Symptome dieser Angriffsstufe zu erzeugen, genügt es, nachdem die Glühbirne durch ein einfaches Ausschalten von der Philips Hue Bridge getrennt wurde, eine weitere ZigBee-Nachricht zu erzeugen. Diese ZigBee-Nachricht kann eine einfache Kopie einer vorherigen Nachricht sein, welche von der Glühbirne ausging und mithilfe des in Kapitel 7.3 beschriebenen Python-Skriptes 'gesammelt' wurde.
Nachdem die ZigBee-Nachricht dupliziert und per logstash-zigbee-in eingelesen wurde, erzeugt das SIEM-System das auf Abbildung 9.3 zu sehende Notable Event. Dabei ist hier neben bereits bekannten Feldern die Adresse der betroffenen Glühbirne als 'bulbAddr' zu sehen.

@timestamp		January 18th 2018, 16:30:16.884
@version		1
_id		AWEJ5F7ARRpyf_JoNRO2
_index		notable-events-index
_score		-
_type		notableEvent
bulbAddr		bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d
effectedEventIDs		_id: AWEJ4elrRRpyf_JoNQsl
message		A Philips Hue bulb is marked as unreachable while still sending zigbee messages. It could be reseted by an attacker.
notableEventtype		Possible Philips Hue Bulb Reset
type		notableEvent

Abbildung 9.3: Notable Event: Factory Reset der Glühbirne (Kibana Screenshot, selbst erstellt)

Glühbirne zu fremdem PAN verbinden Als zweite Angriffsstufe auf dieser Kill Chain Ebene ist die Verbindung der Glühbirne zu einem fremden PAN zu finden. Die Erzeugung der Symptome für den Test der Erkennung dieser Angriffsstufe erfordert komplexere Schritte. Hierbei müssen ZigBee-Nachrichten abgefangen, auf zwei unterschiedlichen Wegen manipuliert und wieder eingeschleust werden.

Um zu simulieren, dass sich die Glühbirne nicht mehr im eigenen PAN befindet, muss das PAN in einer Originalnachricht verändert werden. Dieser Schritt findet in diesem Test statt, nachdem das in 7.3 beschriebene Skript, welches die ZigBee-Daten verfügbar macht, die Daten im JSON-Format abgelegt hat. Im Folgenden sind die entscheidenden Felder einer solchen Nachricht als Ausschnitt der Gesamtnachricht zu sehen. Deutlich zu sehen ist dabei der Unterschied zwischen diesen Rohdaten und den normalisierten Daten, beschrieben in Kapitel 7.5.

```
.
.
"wpan.seqno": "230",
"wpan.dstpan": "0x0000117d",
"wpan.dst16": "0x0000ffff",
"wpan.src16": "0x0000a02d",
"wpan.fcs": "0x00009ae12a",
"wpan.fcsok": "1"
.
.
```

Um die Symptome der Angriffsstufe nachzubilden, muss nun der Eintrag in 'wpan.dstpan', welcher im Zuge der Normalisierung zu 'dest_pan' angepasst wird, verfälscht werden. Hier wird der Eintrag auf '0x0000118d' abgeändert. Damit wird simuliert, dass die Glühbirne ZigBee-Nachrichten in ein fremdes PAN sendet. Wird die Nachricht nun erneut in den Verarbeitungsprozess eingeschleust, wird das auf Abbildung 9.4 sichtbare Notable Event erzeugt.

t	@timestamp	January 18th 2018, 17:28:30.412
t	@version	1
t	_id	AWEKGa1ZgOuAwfnh1lYP
t	_index	notable-events-index
#	_score	-
t	_type	notableEvent
?	bulbAddr	⚠ bulbAddr: 0x0000a02d | bulbAddr: 0x0000a02d
?	bulbAddr2	⚠ bulbAddr:
t	effectedEventIDs	_id: AWEJ9W7agOuAwfnh1gRn | _id: AWEJ9fATgOuAwfnh1gW_
?	effectedEventIDs2	⚠ _id:
?	foreignPAN	⚠ dest_pan: 0x0000118d | dest_pan: 0x0000118d
?	foreignPAN2	⚠ dest_pan:
t	message	A Philips Hue bulb is corresponding with a foreign PAN. It could be compromised and remote controled.
t	notableEventtype	Philips Hue Bulb compromised.
t	type	notableEvent

Abbildung 9.4: Notable Event: Glühbirne wurde zu fremdem PAN verbunden (Kibana Screenshot, selbst erstellt)

Die Erzeugung eines alternativen Symptoms erfordert zusätzlich ein Vertauschen der Feldinhalte 'wpan.dst16' (nach der Normalisierung 'dest_addr') und 'wpan.src16' (nach der Normalisierung 'source_addr'). Dabei kommt es hauptsächlich darauf an, dass die Adresse der Glühbirne in 'wpan.dst16' zu finden ist. Damit wird simuliert, dass sich die Glühbirne bereits in einem fremden PAN befindet und von Nachrichten adressiert wird, welche wiederum ein fremdes PAN adressieren, in welchem sie die Glühbirne erwarten. Auch durch das erneute Einschleusen dieser manipulierten Nachricht wird ein Notable Event wie auf Abbildung 9.4 zu sehen erzeugt.
Mit dem Erzeugen der Notable Events hat das SIEM-System gezeigt, dass es grundsätzlich in der Lage ist, ZigBee-Nachrichten zu verarbeiten und auf deren Basis diese Angriffsstufe zu erkennen.

9.2 Establish a command and control channel

9.2.1 Remote Control Attacke auf Hue Bridge & SSDP Reflection Attacke

Abfluss von Informationen Auf dieser Kill Chain Stufe führen die beiden Angriffsszenarien zunächst einen Abfluss von Informationen per HTTP an einen externen Server des Angreifers durch. Um diesen Fall nachzustellen, muss ein externer Server aus dem LAN per HTTP kontaktiert werden. In diesem Testfall wird als Angreifer Server der Server auf der Blacklist der Use Cases gewählt (siehe Kapitel 8) und vom bereits verwendeten Kali Linux Rechner mit dem Tool cURL[2] angesprochen:

```
1 curl 217.13.68.251
```

Nach der Ausführung des Befehls wird das auf Abbildung 9.5 zu sehende Notable Event im SIEM-System erzeugt. Dabei ist neben den bereits bekannten Feldern die IP-Adresse des internen Gerätes sowie die Ziel-Adresse eingepflegt.

[2] cURL ist ein commandline-basiertes Tool, mit welchem unter anderem Daten per HTTP zu einer URL transportiert werden können (https://curl.haxx.se/)

@timestamp		January 17th 2018, 17:39:38.993
@version		1
_id		AWEE_YT9rSvD7ctSVRPl
_index		notable-events-index
_score		-
_type		notableEvent
effectedEventIDs		_id: AWEE-vV-rSvD7ctSVQMW | _id: AWEE_S-orSvD7ctSVRIB
effectedInternalIPs		source_ip: 192.168.1.243 | source_ip: 192.168.1.243
message		One or multiple hosts send data to suspicious server
notableEventtype		Data flow to suspicious server
suspiciousServerIPs		suspicious_server_ip: 217.13.68.251 | suspicious_server_ip: 217.13.68.251
type		notableEvent

Abbildung 9.5: Notable Event: Abfluss von Informationen (Kibana Screenshot, selbst erstellt)

Mit dem Erzeugen des Notable Events wurde gezeigt, dass das Smart Home SIEM-System diese Angriffsstufe erkennt, sofern sich der Server des Angreifers auf der Blacklist befindet.

Port Mapping Die zweite Angriffsstufe, die sich in Bezug zu diesen beiden Angriffsszenarien auf der Kill Chain Stufe 'Establish a command and control channel' befindet, führt ein SSDP-Port Mapping auf dem Router durch. Um diesen Fall nachzustellen, müssen Flow-Daten erzeugt werden, welche gezielt den SSDP-Port (1900) des Routers der Versuchsumgebung ansprechen. Dies wird abermals mithilfe des Kali Linux Rechners und dem Tool netcat[3] durchgeführt:

```
1 nc -u 192.168.1.1 1900
```

Das SIEM-System erzeugt daraus ein Notable Event mit bereits bekannten Feldern, zu sehen auf Abbildung 9.6.

[3]Netcat ist ein command-line basiertes Tool zum Erzeugen von Netzwerkverbindungen - (https://www.jfranken.de/homepages/johannes/vortraege/netcat.de.html)

@timestamp		January 17th 2018, 17:55:13.691
@version		1
_id		AWEFC8glrSvD7ctSVVe5
_index		notable-events-index
_score		-
_type		notableEvent
effectedEventIDs		_id: AWEFB03BrSvD7ctSVUT4 | _id: AWEFB03BrSvD7ctSVUT9 | _id: AWEFB03BrSvD7ctSVUUE
effectedSourceIPs		source_ip: 192.168.1.243 | source_ip: 192.168.1.243 | source_ip: 192.168.1.243
message		A suspicious device send a ssdp message to the router. This marks a potential port mapping.
notableEventtype		Potential port mapping on router
type		notableEvent

Abbildung 9.6: Notable Event: Port Mapping (Kibana Screenshot, selbst erstellt)

Das Smart Home SIEM-System erkennt somit typische Anzeichen eines SSDP-Port Mappings auf dem Router.

9.3 Accomblish the Task

9.3.1 Remote Control Attacke auf Hue Bridge

Bedienen der Philips Hue Beim Test dieser finalen Angriffsstufe der 'Remote Control Attacke auf Hue Bridge' werden zwei unterschiedliche Varianten getestet. Zum einen soll eine fehlgeschlagene Anmeldung auf der Hue Bridge nachgestellt werden, zum anderen eine erfolgreiche Steuerung der Hue Bridge. Beide Varianten setzen voraus, dass sich der Angreifer außerhalb des LAN befindet und nur dank des zuvor durchgeführten Port Mappings Zugriff auf die API der Philips Hue Bridge hat (siehe Kapitel 6.2.1).
Vorbereitend muss folglich ein tatsächliches Port Mapping auf dem Router durchgeführt werden. Als Ergebnis des Port Mappings ist ein Eintrag in der Firewall-Übersicht der OpenWRT-Oberfläche des Routers zu sehen (siehe Abbildung 9.7). Damit leitet der Routers auf Port 80 erhaltenen Netzwerkverkehr direkt an die API der Philips Hue Bridge weiter (192.168.1.143:80). So sind die Voraussetzungen für den Angriff erfüllt.

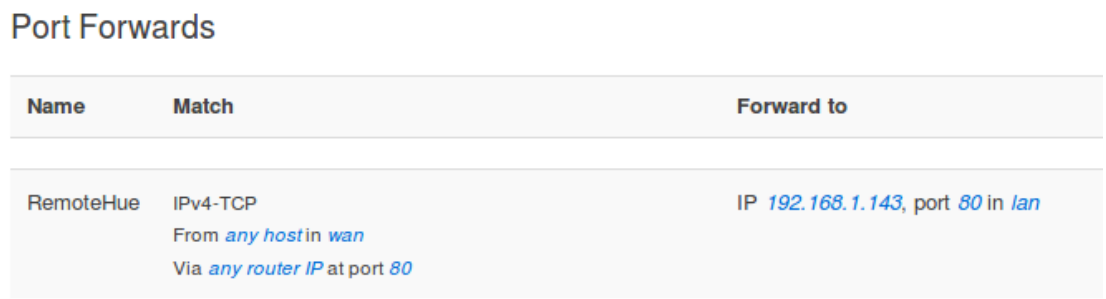


Abbildung 9.7: Port Mapping (OpenWRT Screenshot, selbst erstellt)

Um einen Bedienungsversuch auf die Philips Hue Bridge zu simulieren, wurde das auf Abbildung 9.8 zu sehende, kleine Bash-Skript entworfen. Dabei wird die Philips Hue Bridge API über den gemappten Port auf der WAN-Adresse des Routers angesprochen, jedoch der falsche Benutzername verwendet (der richtige Benutzername ist aus Abbildung 7.4, Feld 'query' bekannt). Das Skript wird erneut auf dem Kali Linux Rechner, welcher sich dieses Mal außerhalb des Smart Home LAN befindet und die IP-Adresse 192.168.178.52 hat, ausgeführt.

```
#!/bin/bash

curl -X PUT http://192.168.178.47:80/api/aksdhawjd1212/lights
```

Abbildung 9.8: Fehlgeschlagenes Bedienen der Philips Hue API (Screenshot, selbst erstellt)

Das SIEM-System produziert nach diesem Angriff das Notable Event auf Abbildung 9.9. Dabei ist neben bereits bekannten Feldern die Anzahl der erfolglosen Versuche innerhalb der letzten 5 Minuten abgebildet ('accessTries'), um Hinweise auf eine mögliche Brute Force Attacke zu erhalten.

@timestamp		January 18th 2018, 14:24:39.347
t	@version	1
t	_id	AWEJcVs_RRpyf_JoMgfP
t	_index	notable-events-index
#	_score	-
t	_type	notableEvent
?	accessTries	⚠ 3
t	effectedEventIDs	_id: AWEJcQqTRRpyf_JoMgZg | _id: AWEJcQ54RRpyf_JoMgZl | _id: AWEJcRZJRRpyf_JoMgbz
t	message	A remote device who knows your Philips Hue device and API tries to access with an unauthorized username. This could be a brute force attack.
t	notableEventtype	Attempted remote access on Philips Hue
?	sourceIPs	⚠ 192.168.178.52, 192.168.178.52, 192.168.178.52
t	type	notableEvent

Abbildung 9.9: Notable Event: Fehlgeschlagenes Bedienen der Philips Hue Bridge (Kibana Screenshot, selbst erstellt)

Zum erfolgreichen Bedienen der Philips Hue Bridge wurde das Angriffsskript erweitert, sodass es ein valides Kommando zum Ändern der Helligkeit der Glühbirne erzeugen kann (siehe Abbildung 9.10). Der ausführende Rechner ist erneut der extern angebundene Kali Linux Rechner.

```
#!/bin/bash

brightness=$1
on=$2

curl -X PUT -H "Content-Type: application/json" -d '{"on":'$on',"bri":'$brightness'}'
http://192.168.178.47:80/api/x9Z-zi5yhvUMwwAFxIKUq3jg6Vd1cuW-H5gpmSvL/lights/2/state
```

Abbildung 9.10: Erfolgreiches Bedienen der Philips Hue API (Screenshot, selbst erstellt)

Das Smart Home SIEM-System erzeugt das Notable Event auf Abbildung 9.11, auf dem neben den bereits bekannten Feldern das Feld 'executedActions' mit der von OpenHAB gemeldeten Änderung des Zustandes enthalten ist.

@timestamp		January 18th 2018, 14:22:14.974
t	@version	1
t	_id	AWEJbydlRRpyf_JoMf7W
t	_index	notable-events-index
#	_score	-
t	_type	notableEvent
t	effectedEventIDs	_id: AWEJbjvQRRpyf_JoMfoY | _id: AWEJbnZpRRpyf_JoMfrA
?	effectedEventIDs2	⚠ _id: AWEJbkCXRRpyf_JoMfoa | _id: AWEJbo7MRRpyf_JoMfto
?	executedActions	⚠ message: 2018-01-18 14:21:15.049 [ItemStateChangedEvent] - hue_0100_00178870c72d_2_brightness changed from 100 to 39 | message: 2018-01-18 14:21:35.049 [ItemStateChangedEvent] - hue_0100_00178870c72d_2_brightness changed from 39 to 59
t	message	A remote device did sucessfully enter the Philips Hue API and changed the state of the light bulbs.
t	notableEventtype	Successful remote control attack on Philips Hue
?	sourceIPs	⚠ source_ip: 192.168.178.52 | source_ip: 192.168.178.52
t	type	notableEvent

Abbildung 9.11: Notable Event: Erfolgreiches Bedienen der Philips Hue Bridge (Kibana Screenshot, selbst erstellt)

Beide Angriffsfälle dieser Angriffsstufe wurden damit vom SIEM-System erkannt.

9.3.2 SSDP Reflection Attacke

Da die SSDP Reflection Attacke, wie in Kapitel 6 beschrieben, das Smart Home auf zwei unterschiedlichen Wegen treffen kann, sind zwei unterschiedliche Tests des Systems notwendig. Um ein hohes Aufkommen an SSDP-Nachrichten zu simulieren und damit zu prüfen, ob eine Smart Home Komponente selbst Opfer einer solchen Attacke ist, wurde folgender Befehl innerhalb des LAN mehrfach ausgeführt und damit die Philips Hue Bridge adressiert.

```
1 nc -u 192.168.1.143 1900
```

Als Resultat wurde das Notable Event auf Abbildung 9.12 erzeugt.

t	@timestamp	January 17th 2018, 18:40:49.271
t	@version	1
t	_id	AWEFNYYIrSvD7ctSVij7
t	_index	notable-events-index
#	_score	-
t	_type	notableEvent
?	destIPs	dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143 | dest_ip: 192.168.1.143
t	effectedEventIDs	_id: AWEFMUPzrSvD7ctSVhZ6 | _id: AWEFMUPzrSvD7ctSVhZ- | _id: AWEFMUPzrSvD7ctSVhZ_ | _id: AWEFMUPzrSvD7ctSVhaJ | _id: AWEFMRzerSvD7ctSVhXU | _id: AWEFMRzerSvD7ctSVhXa | _id: AWEFMRzerSvD7ctSVhXb | _id: AWEFMRzerSvD7ctSVhXe | _id: AWEFMRzerSvD7ctSVhXg | _id: AWEFMPXirSvD7ctSVhVi
t	message	There is an unusual high ssdp traffic in the LAN. That could be a mark of a ongoing ssdp reflection attack.
t	notableEventtype	Possible SSDP Reflection Attack detected
?	sourceIPs	source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189
t	type	notableEvent

Abbildung 9.12: Notable Event: SSDP Reflection Attacke 1 (Kibana Screenshot, selbst erstellt)

In der zweiten Form dieser Angriffsstufe versendet eine eigene Smart Home Komponente selbst SSDP-Nachrichten an ein externes Gerät (siehe Kapitel 6). Um diesen Fall zu testen, wird folgender Befehl von innerhalb des LAN an eine externe Adresse ausgeführt:

```
1 nc -u 192.168.178.52 1900
```

Auch hier erzeugt das SIEM-System ein Notable Event, welches auf Abbildung 9.13 zu sehen ist.

t	@timestamp	January 17th 2018, 19:32:11.089
t	@version	1
t	_id	AWEFZIxarSvD7ctSVrC2
t	_index	notable-events-index
#	_score	-
t	_type	notableEvent
?	destIPs	dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52 | dest_ip: 192.168.78.52
t	effectedEventIDs	_id: AWEFYN9urSvD7ctSVqTK | _id: AWEFY1BjrSvD7ctSVqxM | _id: AWEFYLhhrSvD7ctSVqQz | _id: AWEFYQaDrSvD7ctSVqVE | _id: AWEFY8W2rSvD7ctSVq5E | _id: AWEFY-y9rSvD7ctSVq7D | _id: AWEFY3d2rSvD7ctSVq0k | _id: AWEFYVSTrSvD7ctSVqZk | _id: AWEFY56UrSvD7ctSVq2p | _id: AWEFYS2QrSvD7ctSVqXZ
t	message	There is SSDP traffic to the outside of the LAN. This could be a mark of a ongoing SSDP Reflection Attack.
t	notableEventtype	Possible SSDP Reflection Attack detected.
?	sourceIPs	source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189 | source_ip: 192.168.1.189
t	type	notableEvent

Abbildung 9.13: Notable Event: SSDP Reflection Attacke 2 (Kibana Screenshot, selbst erstellt)

Damit wurden typische Anzeichen dieser Angriffsstufe vom SIEM-System erkannt.

9.4 Zusammenfassung

Nach der Durchführung der Tests konnte gezeigt werden, dass die ausgewählten Angriffsstufen der Kill Chain grundsätzlich vom hier entwickelten SIEM Ansatz erkannt werden können. Sowohl die OpenHAB Status Events als auch die ZigBee-Daten können, mit allen bereits erwähnten Einschränkungen der Tests, interpretiert werden und führen zum Anschlagen der dafür entwickelten Use Cases.

10 Fazit

Das Ziel dieser Arbeit war es, eine mögliche Übertragung des SIEM Ansatzes aus dem unternehmerischen Umfeld und großen IT-Infrastrukturen auf Smart Home Umgebungen zu prüfen. Dabei lag der Fokus auf der technischen Machbarkeit, wobei die operativen Aspekte nur am Rande betrachtet wurden. Dieses Kapitel soll als Fazit der Entwicklungen und Resultate dienen. Dabei wird ein Resümee über die einzelnen Schritte dieser Arbeit sowie die Funktionalität des entwickelten SIEM-Systems gezogen. Außerdem wird ein Blick auf den für das Smart Home SIEM-System benötigten Aufwand geworfen, um die Umsetzbarkeit auf reale Umgebungen zu bewerten. Abschließend wird die Übertragbarkeit des Ansatzes auf andere Smart Home Umgebungen, die von der hier verwendeten Testumgebung abweichen, eingeschätzt.
Im Zuge dieses Fazits werden auch die bislang nur nebensächlich behandelten operativen Aspekte verstärkt ins Visier genommen. Einen Umriss möglicher zukünftiger Arbeiten an dieser Thematik bietet das darauf folgende Kapitel 11 an.

Vorgehensweise, Implementierung & Funktionalität Nach der Einleitung (Kapitel 1), welche neben einer Beschreibung der aktuellen Situation und der Motivation, sich mit dieser Thematik zu beschäftigen, eine Herleitung der Notwendigkeit der Übertragung des SIEM Ansatzes ins Smart Home aufzeigt, wurde in Kapitel 2 eine Beschreibung der Grundlagen für das Verständnis von SIEM-Systemen und deren Einordnung in den Protect-Detect-React Ansatz vorgenommen.
Um eine strukturierte Herangehensweise dieser Übertragung und eine präzisere Zielsetzung zu ermöglichen, wurde nach einer Grundlagenbeschreibung des IoT und des Smart Home als ein Teilbereich dessen, in Kapitel 3 eine erste theoretische Übertragung des Ansatzes in dieses Umfeld erarbeitet. Dazu wurde das Smart Home zunächst mit der 3-Layer-Architektur des IoT in Zusammenhang gebracht, als Voraussetzung für das Ziel, mit dem SIEM Ansatz Angriffe oder Angriffsvorbereitungen auf allen drei IoT-Layern erkennen zu können. Im Anschluss wurden Herausforderungen dargestellt, mit der diese Übertragung zu kämpfen hat. Nach einer Übersicht über unterschiedliche, bereits bestehende Ansätze wurde im Folgekapitel 4 schließlich ein eigener Ansatz als eine Kombination vorhandener Arbeiten und eigener Überlegungen

skizziert.
In Kapitel 5 wurde eine Smart Home Testumgebung beschrieben, welche nach Vorbildern in der Literatur sowie Voraussetzungen aus der theoretischen Skizze des eigenen Ansatzes (bspw. des Einsatzes von OpenHAB) entwickelt wurde. Auch erste Maßnahmen für das spätere Sammeln von Daten mussten an dieser Stelle umgesetzt werden, indem der Router mit OpenWRT bespielt und der ZigBee-Sniffer positioniert wurde. Dabei wurde festgestellt, dass es, entgegen erster Annahmen, ausreicht, eine einzige Smart Home Komponente, die Philips Hue, einzusetzen, um die Übertragung des Ansatzes exemplarisch zeigen zu können. Die Versuchsumgebung wurde in die zuvor erörterte 3-Layer-Architektur des Smart Home übertragen, um die Verfolgung der Zielsetzung, Angriffe auf allen drei Layern zu erkennen, weiterführen zu können.
Als weiterer Bestandteil der Zielsetzung ist die im Kapitel 6 erfolgte Auswahl und Aufarbeitung unterschiedlicher Angriffsszenarien zu betrachten. Hier wurden drei Angriffsszenarien ausgewählt, in Kill Chains aufgeteilt und in theoretischen Use Cases strukturiert. Dabei konnte durch die zur Erkennung benötigten Daten gezeigt werden, dass die einzelnen Stufen der Angriffsszenarien auf alle drei IoT-Layer zielen und damit den Anforderungen einer ganzheitlichen Betrachtung des Smart Home genügen. Aufgrund der benötigten Daten wurden einige Use Cases aussortiert und nicht weiter verwendet, da sie außerhalb des hier angedachten SIEM Ansatzes liegen (bspw. Prozesse auf Endgerät überwachen). Hierfür müssten Methoden eingesetzt werden, die von einem herkömmlichen SIEM-System (siehe Abbildung 2.2) ebenfalls nicht verwendet werden. Eine Abdeckung der ersten Kill Chain Stufen ist mit dem SIEM Ansatz ebenfalls nicht möglich, da sie außerhalb des physischen Einzugsbereiches (außerhalb des LAN) geschehen. Dadurch bleiben diverse Vorbereitungsstufen dieser Angriffe unentdeckt, was jedoch eine grundsätzliche Problematik des SIEM Ansatzes beschreibt und nicht auf das Smart Home SIEM beschränkt ist.
Die technische Umsetzung des SIEM Ansatzes wurde in Kapitel 7 mit der Beschreibung der Implementierung des Systems, welches alle in dieser Arbeit geforderten funktionalen Schritte eines SIEM-Systems abdeckt, festgehalten. Die Voraussetzungen für formulierte Anforderungen wie die notwendige Skalierbarkeit und Erweiterbarkeit wurden dabei geschaffen.
Die Implementierung der Use Cases mithilfe von Python und der Elasticsearch Query DSL ist in Kapitel 8 dokumentiert. Dabei wurden neben den erwartbaren Herausforderungen auch Probleme mit der Elasticsearch Query DSL deutlich. So wurde bspw. kein einfacher, sauberer Weg gefunden, IP-Adressbereiche von Suchen auszuschließen. Dennoch ist es gelungen, die in Kapitel 6.3 vorgegebenen Use Cases umzusetzen.
Im Anschluss wurde die Funktionalität des SIEM-Systems in Kapitel 9 mit dem Ziel getestet, die Angriffsstufen (Kill Chains, siehe Abbildung 6.1) der einzelnen Angriffsszenarien zu erkennen.

Dabei gelang es im Fall der ZigBee-Angriffe lediglich die Symptome der Angriffe nachzustellen, nicht jedoch die Angriffsstufen real auszuführen oder echte ZigBee-Nachrichten zu erzeugen, da dies zu nicht überschaubarem Mehraufwand und einer Verlagerung des Fokus dieser Arbeit geführt hätte. Die anderen Angriffsszenarien konnten unter besseren Bedingungen nachgestellt werden. Durch die Tests wurde gezeigt, dass die umgesetzten Use Cases die ausgewählten Angriffsszenarien auf entscheidenden Stufen erkennen und es damit gelungen ist, den SIEM Ansatz technisch, funktional und unter Verwendung von Open Source Komponenten auf das Smart Home zu übertragen und eine Abdeckung aller drei IoT-Layer zu erreichen.

Aufwand & Umsetzbarkeit Um ein Fazit über die in dieser Arbeit durchgeführte Übertragung zu ziehen, ist neben der zuvor erfolgten Prüfung der Funktionalität des Systems auch der Aspekt des dazu nötigen Aufwandes zu betrachten, da dieser ein entscheidender Faktor für die Umsetzbarkeit in realen Umgebungen von Smart Home Besitzern ist. Das gilt insbesondere für den Aufwand außerhalb bzw. nach der Entwicklung des Systems.
In die unterschiedlichen Implementierungsstufen der Umsetzung muss viel Aufwand investiert werden, gepaart mit weitreichendem notwendigem Know-How. Das liegt zum einen an den bereits in Kapitel 3.3 angesprochenen Problemen. Besonders die Erhebung und Normalisierung benötigter Log-Daten, die in herkömmlichem SIEM-Systemen keine Verwendung finden, ist herausfordernd. Wo die Flow- und HTTP-Daten, also Daten des Application und des Transport Layers, noch mit relativ einfachen Mitteln einzubinden waren, entstand bei der Sammlung der ZigBee-Daten eine komplexere Aufgabe, welche einige Workarounds und zusätzlich verfasste Skripte erforderte. Auch bei der Normalisierung lag der größte Aufwand bei der Aufbereitung der ZigBee-Daten aber auch der OpenHAB Status Events.
Die Implementierung der Use Cases setzt ebenfalls eine tiefe Beschäftigung mit einzelnen Angriffsszenarien voraus. Dabei ist zu bemerken, dass innerhalb dieser Arbeit lediglich drei Angriffsszenarien betrachtet wurden. Soll hierbei ein ganzes Spektrum an aktuellen Angriffen abgedeckt werden, wäre eine kontinuierliche, weitreichende Beschäftigung mit ihnen erforderlich. Doch auch nach der erfolgreichen Implementierung der Use Cases ist für die Analyse und Bewertung der Notable Events sowie für die Einleitung von Gegenmaßnahmen, welche ebenfalls zu definieren sind, tiefes Verständnis vonnöten.
Aus den hier gezeigten Schwierigkeiten folgt die Grundproblematik dieses Ansatzes. Er erfordert auch nach der Entwicklung und dem initialen Aufsetzen einen Aufwand und ein Know-How, das für durchschnittliche Smart Home Besitzer nicht zu leisten ist. Ansätze und notwendige Maßnahmen, um sich einem Zustand zu nähern, in dem ein Smart Home SIEM-

System real verwendbar und operativ betreibbar wäre, werden in Kapitel 11 gezeigt.

Übertragbarkeit auf andere Smart Home Umgebungen Ein weiterer Gesichtspunkt ist die mögliche Übertragbarkeit auf andere Smart Home Umgebungen. Im Rahmen dieser Arbeit wurde ein SIEM Ansatz innerhalb einer Versuchsumgebung entwickelt, die, wie in Kapitel 5 beschrieben, ein typisches Smart Home darstellt. Jedoch ist aufgrund unterschiedlicher Kriterien eine Übertragbarkeit teilweise eingeschränkt bzw. an Bedingungen geknüpft. So ist der eigene Ansatz (Kapitel 4) auf die von OpenHAB gelieferten Status Events angewiesen. Damit können nur von OpenHAB unterstützte Smart Home Komponenten angebunden werden. OpenHAB unterstützt bereits eine Vielzahl populärer Komponenten und wird stetig mit weiteren dazu benötigten Schnittstellen ausgestattet [Meffert u. a., 2017]. Darunter befinden sich auch die bereits in Kapitel 5 erwähnten Geräte Belkin WeMo und der Nest smoke-alarm, für die ebenfalls bereits Schwachstellen gezeigt wurden [Notra u. a., 2014]. Ein großer Teil der typischen Smart Home Umgebungen könnte damit abgedeckt werden. Wird ein Smart Home Gerät von OpenHAB unterstützt, müsste für dessen Einbindung unter Umständen der Python-Code der parser Komponente (siehe Abbildung 7.1) angepasst, jedoch keine grundlegenden Änderungen vorgenommen werden. Damit wäre der Aufwand an dieser Stelle überschaubar. Ist keine OpenHAB Unterstützung für die Smart Home Geräte vorhanden, würden wichtige Erkenntnisse verloren gehen und entscheidende Angriffsstufen nicht zuverlässig erkannt werden.
Problematischer ist die Einbeziehung weiterer Datenquellen aus dem Perception Layer. Wird statt ZigBee eine andere Low-Energy Übertragungsmethode (bspw. Bluetooth) der IoT-Objekte zu ihrem Gateway verwendet, müsste unter Umständen eine ähnliche Methode wie die bei der Einbindung der ZigBee-Daten innerhalb dieser Arbeit (siehe Kapitel 7.3) entwickelt werden, was mit größerem Aufwand verbunden wäre. Nachdem das Erschließen der Daten erfolgreich abgeschlossen ist und diese in Elasticsearch angekommen sind, wären die weiteren Schritte, wie die Normalisierung der Daten, unproblematisch.
Das Aufsetzen des SIEM-Systems selbst ist aufgrund seiner Docker-Architektur in anderen Smart Homes mit überschaubarem Aufwand zu leisten. Jedoch müssen für jede Smart Home Umgebung die Geräteliste (Abbildung 5.1) und damit auch die Use Cases (Kapitel 8) angepasst werden. Auch Konfigurationen, wie bspw. bei der Einbindung der HTTP- und Flow-Daten, müssen vorgenommen werden. Zusätzlich muss der Router eine Weiterleitung der Netzwerkdaten unterstützen oder wie in dieser Arbeit mit OpenWRT ausgestattet werden, was zusätzlichen Aufwand bedeutet. Insgesamt sind für einen SIEM Einsatz damit starke Anpassungen in der Smart Home Umgebung notwendig.

Die Entwicklung neuer Use Cases, welche Angriffsszenarien auf Smart Home Komponenten anderer Umgebungen aufdecken sollen, wäre, vorausgesetzt die Daten wurden erfolgreich gesammelt und normalisiert, möglich. Der Aufwand wäre analog zur Use Case Entwicklung innerhalb dieser Arbeit zu betrachten.

Aufgrund der gezeigten Notwendigkeit der Betrachtung von Detect Mechanismen im Smart Home, ist eine Beschäftigung mit möglichen weiterführenden Arbeiten trotz oder gerade wegen der Menge an Herausforderungen lohnenswert. Diese Arbeit hat mit der konkreten Umsetzung des SIEM Ansatzes im Smart Home und der gezeigten Funktionalität einen der notwendigen Grundsteine dafür gelegt.

11 Ausblick

Um den SIEM Ansatz im Smart Home praktikabel zu machen und über das Stadium eines experimentellen Prototypen hinaus zu entwickeln, sind unterschiedliche Maßnahmen möglich und nötig, die im Folgenden erläutert werden. Hierbei wird grundsätzlich zwischen weiteren Arbeiten aus wissenschaftlicher und experimenteller Perspektive und notwendigen Maßnahmen von Seiten der Hersteller von Smart Home Komponenten oder möglichen Smart Home SIEM-Systemen unterschieden.
Zunächst sind konkrete nächste Schritte bei der Weiterentwicklung der technischen Implementierung des SIEM-Systems zu nennen. Das in Kapitel 7 entwickelte Gesamtsystem müsste innerhalb zukünftiger Projekte in Bezug auf viele Funktionalitäten robuster gemacht und verfeinert werden. Dazu ist eine ausgedehnte, strukturierte Testphase besonders notwendig, um Fehler im Gesamtsystem zu finden und die Zuverlässigkeit für einen dauerhaften Betrieb zu gewährleisten. Dabei steht auch die Zusammenarbeit der unterschiedlichen Microservices im Fokus. Auch die in Kapitel 2.2.3 erwähnte Security des SIEM-Systems selbst ist zu testen. Die Use Cases gilt es auf Basis der beim Testen gesammelten Erfahrungen zu verbessern. False Positives müssen verringert und nicht erkannte Angriffsstufen verhindert werden. Auch müssten komplexere Penetrationtests entwickelt werden, welche die Use Cases unter realen Bedingungen, insbesondere auf der ZigBee-Ebene, also dem Perception Layer, prüfen.
Desweiteren muss das SIEM-System um zusätzliche Funktionalitäten erweitert werden. So ist die Priorisierung von Notable Events, welche in [Aman und Snekkenes, 2014] beschrieben ist und auch in Kapitel 2 Erwähnung findet, ein wichtiger Aspekt, welcher im SIEM-System mithilfe zusätzlicher Funktionalität möglich gemacht werden muss. Auch müssen weitere Log-Quellen angeschlossen, weitere Protokolle des Perception Layer eingebunden und normalisiert werden. Beispielhaft sind Log-Daten des Routers zu nennen, welche bspw. über Anmeldeversuche oder illegalerweise vorgenommene Konfigurationen Auskunft geben. Die Integration eines Threat Intelligence Feeds, welcher das SIEM-System bspw. mit aktuellen Blacklists versorgt, ist ebenfalls eine denkbare Fortsetzung der Arbeit. Mit einem in Kapitel 2 erwähnten Schwachstellenscanner könnte eine Übertragung ins Smart Home geprüft werden, um ihn anschließend als zusätzliche Log-Quelle an das Smart Home SIEM-System anzubinden.

Dadurch könnten aktuelle Angriffsversuche mit gefundenen Schwachstellen der Geräte korreliert werden, um die Kritikalität der Angriffe zu bewerten. Als Erkennungsziel für solche Erweiterungen könnten entsprechende Angriffsszenarien aus der in Kapitel 1 aufgeführter Literatur herangezogen werden.
Verfeinert und erweitert werden muss auch die Normalisierung, Aufarbeitung und Darstellung der Daten. So müssen die gesammelten Daten verstärkt nach Notwendigkeit gefiltert, aufbereitet und in zusätzlich entwickelten Kibana Dashboards, welche dem Benutzer auf einfache Art und Weise einen visuellen Überblick über die aktuelle Situation verschaffen, dargestellt werden. Auch Darstellung und Inhalt der Notable Events sind zu verbessern, indem weitere wertvolle Informationen wie bspw. aufbereitete, dem Notable Event vorangegangene Aktivitäten der beteiligten Systeme zur Analyse hinzu- und bessere, strukturierte Verlinkungen zu den auslösenden Events eingeführt werden.

Um den Smart Home SIEM Ansatz einsatzfähig und auf unterschiedlichste Smart Home Umgebungen übertragbar zu machen, sind insbesondere auch Hersteller von Smart Home Geräten gefragt. Sie müssten die gängigen Probleme, siehe Kapitel 3.3 und 10, überwinden, indem die Geräte zugänglich gemacht werden sowie das Logging der Geräte und Anwendungen eingebaut oder erweitert und dokumentiert wird. Dies gilt insbesondere für den Perception Layer. Würden Status Event Logs von Smart Home Geräten selbst erzeugt werden, wäre ein Einsatz von OpenHAB für den SIEM Ansatz überflüssig. Gleichzeitig wäre es wichtig, die Vielzahl der verwendeten Low-Energy Protokolle weiter zu standardisieren, um eine Anbindung und Normalisierung der Daten überschaubarer und einheitlicher gestalten zu können.
Einer der wichtigsten Fragenkomplexe, die es zu klären gilt, beinhaltet die in dieser Arbeit nur nebensächlich behandelten operativen Aspekten. Fortführende Arbeiten müssten diese Gesichtspunkte in den Fokus nehmen, da sie essentiell für den Einsatz des SIEM Ansatzes sind. Darin sind im Folgenden gezeigte Fragen verortet. Wer entwickelt und pflegt Use Cases und stellt sie zusammen mit weiteren Updates dem Smart Home SIEM-System zur Verfügung? Dies könnten die Hersteller der Geräte, ein Threat Intelligence Feed Anbieter oder der Hersteller des SIEM-Systems sein. Wer überwacht das SIEM-System und analysiert Notable Events? Hier müsste eine zentrale Lösung entworfen werden, da nicht jedes Smart Home sein eigenes SOC beschäftigen kann. Schließlich stellt sich auch die Frage nach dem Incident Response Prozess: Wer leitet Gegenmaßnahmen ein, wenn Notable Events erzeugt und analysiert wurden? Ein möglicher Grundansatz für Hersteller könnte ein neues Geschäftsmodell sein, welches nicht nur auf der verkauften Stückzahl der Smart Home Geräte, sondern auch auf einem damit einhergehenden SIEM-Service (oder Security-Service allgemein) basiert. So könnte der in

Kapitel 3.3 angesprochenen Problematik, dass Hersteller ihren Fokus aufgrund des aktuellen Geschäftsmodelles nicht auf Wartung, Patches, usw. legen, begegnet werden. Diese Fragen führen zwangsläufig auch zur Beschäftigung mit der Thematik des Datenschutzes, da die von einem SIEM-System gesammelten Daten personenbezogen sein können oder Schlüsse auf das Verhalten von Smart Home Bewohnern zulassen. Wer darf welche Daten sehen, wenn ein solches SIEM-System zentral verwaltet wird?
Um den SIEM Ansatz im Smart Home weiter zu verfolgen, sind sowohl von wissenschaftlicher als auch von der Herstellerseite große Anstrengungen gefordert. Diese müssen aber bemüht werden und sei es nur, um den Smart Home SIEM Ansatz aufgrund von entscheidenden Kriterien zu verwerfen und Alternativen zu entwickeln. Die Notwendigkeit, Detect Mechanismen in einer vernetzten Privatwelt auf das Smart Home im Speziellen und IoT-Geräte im Allgemeinen auszuweiten, ist gegeben.

Literaturverzeichnis

[AkamaiTechnologies 2014] AkamaiTechnologies: SSDP Reflection DDoS Attacks. (2014), Oktober. – URL https://www.akamai.com/uk/en/multimedia/documents/state-of-the-internet/ssdp-reflection-ddos-attacks-threat-advisory.pdf

[Alaa u. a. 2017] Alaa, Mussab ; Zaidan, A ; Zaidan, B.B. ; Talal, Mohammed ; Mat Kiah, Miss L.: A Review of Smart Home Applications based on Internet of Things. 97 (2017), 09

[Alam u. a. 2012] Alam, M. R. ; Reaz, M. B. I. ; Ali, M. A. M.: A Review of Smart Homes - Past, Present, and Future. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42 (2012), Nov, Nr. 6, S. 1190–1203. – ISSN 1094-6977

[AlienVault] AlienVault: *AlienVault OSSIM: The World's Most Widely Used Open Source SIEM*. – URL https://www.alienvault.com/products/ossim. – eingesehen: 08.12.2017

[Allerding und Schmeck 2011] Allerding, Florian ; Schmeck, Hartmut: Organic Smart Home: Architecture for Energy Management in Intelligent Buildings. In: *Proceedings of the 2011 Workshop on Organic Computing*. New York, NY, USA : ACM, 2011 (OC '11), S. 67–76. – URL http://doi.acm.org/10.1145/1998642.1998654. – ISBN 978-1-4503-0736-9

[Aman und Snekkenes 2014] Aman, Waqas ; Snekkenes, Einar: Event Driven Adaptive Security in Internet of things. (2014), 08

[Apthorpe u. a. 2017a] Apthorpe, Noah ; Reisman, Dillon ; Feamster, Nick: Closing the Blinds: Four Strategies for Protecting Smart Home Privacy from Network Observers. In: *CoRR* abs/1705.06809 (2017). – URL http://arxiv.org/abs/1705.06809

[Apthorpe u. a. 2017b] Apthorpe, Noah ; Reisman, Dillon ; Feamster, Nick: A Smart Home is No Castle: Privacy Vulnerabilities of Encrypted IoT Traffic. In: *CoRR* abs/1705.06805 (2017). – URL http://arxiv.org/abs/1705.06805

[Atzori u. a. 2010] Atzori, Luigi ; Iera, Antonio ; Morabito, Giacomo: The Internet of Things: A Survey. In: *Comput. Netw.* 54 (2010), Oktober, Nr. 15, S. 2787–2805. – URL http://dx.doi.org/10.1016/j.comnet.2010.05.010. – ISSN 1389-1286

[Blackwell 2009] Blackwell, Clive: A Security Architecture to Protect Against the Insider Threat from Damage, Fraud and Theft. In: *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies*. New York, NY, USA : ACM, 2009 (CSIIRW '09), S. 45:1–45:4. – URL http://doi.acm.org/10.1145/1558607.1558659. – ISBN 978-1-60558-518-5

[BSI 2013] BSI: Studie über die Nutzung von Log- und Monitoringdaten im Rahmen der IT-Frühwarnung und für einen sicheren IT-Betrieb. (2013)

[BSI 2016] BSI: *The State of IT Security in Germany 2016*. 2016

[Bundestag 2017] Bundestag: *Straßenverkehrsgesetz für automatisiertes Fahren geändert*. 2017. – URL https://www.bundestag.de/dokumente/textarchiv/2017/kw13-de-automatisiertes-fahren/499928. – eingesehen am: 26.01.2018

[Butler 2009] Butler, Michael: Benchmarking Security Information Event Management (SIEM). In: *SANS Institute Reading Room* (2009). – URL https://www.sans.org/reading-room/whitepapers/analyst/benchmarking-security-information-event-management-siem-34755

[Cappelli u. a. 2012] Cappelli, Dawn ; Moore, Andrew ; Trzeciak, Randall: *The CERT Guide to Insider Threats: How to Prevent, Detect, and Respond to Information Technology Crimes (Theft, Sabotage, Fraud)*. Addison-Wesley Professional, 2012

[Community 2017] Community, SmartThings: *Hue Bulb Factory Reset: SOLVED if your Hue Bulb is on a ZLL Channel*. 2017. – URL https://community.smartthings.com/t/hue-bulb-factory-reset-solved-if-your-hue-bulb-is-on-a-zll-channel/84967. – eingesehen am: 26.01.2018

[Daniel Mahrenholz 2016] Daniel Mahrenholz, Ralf S.: Open-Source-SIEM im Eigenbau. In: *P. Schartner, P. Lipp. 'DACH Security 2016' syssec (2016)* (2016). – URL https://rt-solutions.de/de/2017/01/open-source-siem-im-eigenbau/

[Davidson 2017] Davidson, Ray: Integrating Prevention, Detection and Response Workflows: SANS Survey on Security Optimization. (2017). – URL https://www.sans.org/reading-room/whitepapers/analyst/

integrating-prevention-detection-response-work-flows-survey-security-optimization-37730

[Eckhardt 2017] Eckhardt, Tim: *Security-Monitoring von Internet-of-Things Endgeräten in lokalen Netzwerken unter Einsatz eines intelligenten Systems.* 2017

[enisa 2016] enisa: Major DDoS Attacks Involving IoT Devices. (2016). – URL https://www.enisa.europa.eu/publications/info-notes/major-ddos-attacks-involving-iot-devices

[Fox 2016] Fox, Michael A.: Why is home so important to us? In: *Academic Insights for the Thinking World* (2016). – URL https://blog.oup.com/2016/12/home-place-environment/

[Frye 2009] Frye, Daniel: Effective Use Case Modeling for Security Information and Event Management. In: *SANS Institute Reading Room* (2009). – URL https://www.sans.org/reading-room/whitepapers/logging/effective-case-modeling-security-information-event-management-33319

[Gartner] Gartner: *Gartner 2017 Magic Quadrant for SIEM.* – URL https://www.splunk.com/en_us/form/gartner-siem-magic-quadrant.html. – eingesehen: 01.02.2018

[Gheorghe 2016] Gheorghe, Alexandra: Security Awareness in the Age of Internet of Things. (2016). – URL http://download.bitdefender.com/resources/files/News/CaseStudies/study/136/Bitdefender-Whitepaper-IoTSecurity-A4-en-EN-web.pdf

[Gray und Perniola 2016] Gray, David ; Perniola, Angelo: Targeted SOC Use Cases for Effective mIncident Detection and Response. In: *SANS DFIR SUMMIT PRAGUE* (2016). – URL https://www.sans.org/summit-archives/file/summit-archive-1492186603.pdf

[Handelsblatt] Handelsblatt: *Autonome Taxen Ã¤ndern den Stadtverkehr.* – URL http://www.handelsblatt.com/auto/nachrichten/selbstfahrende-autos-autonome-taxen-aendern-den-stadtverkehr/20803738.html. – eingesehen: 01.02.2018

[Hüning u. a. 2016] Hüning, Christian ; Adebahr, Mitja ; Thiel-Clemen, Thomas ; Dalski, Jan ; Lenfers, Ulfia ; Grundmann, Lukas: Modeling Simulation As a Service with the Massive

Multi-agent System MARS. In: *Proceedings of the Agent-Directed Simulation Symposium*. San Diego, CA, USA : Society for Computer Simulation International, 2016 (ADS '16), S. 1:1–1:8. – URL http://dl.acm.org/citation.cfm?id=2972193.2972194. – ISBN 978-1-5108-2315-0

[IDC 2017] IDC: IDC Studie zu Next Gen Endpoint Security in deutschen Unternehmen: Gefahr erkannt, aber nicht gebannt. (2017). – URL http://idc.de/de/ueber-idc/press-center/64840-idc-studie-zu-next-gen-endpoint-security-in-deutschen-unternehmen-gefahr-erkannt-aber-nicht-gebannt

[ISACA 2016] ISACA: State of Cybersecurity: Implicactions for 2016. In: *ISACA and RSA Conference* (2016). – URL http://www.isaca.org/cyber/Documents/state-of-cybersecurity_res_eng_0316.pdf

[Jacobsson u. a. 2014] Jacobsson, A. ; Boldt, M. ; Carlsson, B.: On the Risk Exposure of Smart Home Automation Systems. In: *2014 International Conference on Future Internet of Things and Cloud*, Aug 2014, S. 183–190

[Jäger 2015a] Jäger, Thomas: *Intrusion Detection System auf einem Single Board Computer*, HAW Hamburg, Bachelorarbeit, 2015

[Jäger 2015b] Jäger, Thomas: *Safety and Security*. 2015

[Jäger 2017] Jäger, Thomas: *Security von Import-Diensten im Docker-Umfeld*. 2017

[Jangla u. a. 2015] Jangla, Gurpreet ; Kaur, Amne ; Deepa.A.: Development of an Intrusion Detection System based on Big Data for Detecting Unknown Attacks. (2015). – URL http://www.ijarcce.com/upload/2015/december-15/IJARCCE%2052.pdf

[Jing u. a. 2014] Jing, Qi ; Vasilakos, Athanasios V. ; Wan, Jiafu ; Lu, Jingwei ; Qiu, Dechao: Security of the Internet of Things: Perspectives and Challenges. In: *Wirel. Netw.* 20 (2014), November, Nr. 8, S. 2481–2501. – URL http://dx.doi.org/10.1007/s11276-014-0761-7. – ISSN 1022-0038

[Kodali u. a. 2016] Kodali, Ravi ; Jain, Vishal ; Bose, Suvadeep ; Boppana, Lakshmi: IoT based smart security and home automation system. (2016), 04, S. 1286–1289

[Kon u. a. 2017] Kon, Bethany ; Lam, Alex ; Chan, Jonathan: Evolution of Smart Homes for the Elderly. In: *Proceedings of the 26th International Conference on World Wide Web*

Companion. Republic and Canton of Geneva, Switzerland : International World Wide Web Conferences Steering Committee, 2017 (WWW '17 Companion), S. 1095–1101. – URL https://doi.org/10.1145/3041021.3054928. – ISBN 978-1-4503-4914-7

[Koramis 2017] Koramis: Project Haunted House. (2017). – URL https://www.sophos-events.com/smarthome/en/form.cfm. – eingesehen am: 26.01.2018

[LaPiedra 2002] LaPiedra, James: The Information Security Process Prevention, Detection and Response. (2002)

[Lavrova und Pechenkin 2015] Lavrova, Daria ; Pechenkin, Alexander: Applying Correlation and Regression Analysis to Detect Security Incidents in the Internet of Things. In: *International Journal of Communication Networks and Information Security (IJCNIS)* 7 (2015), Dezember, Nr. 3, S. 131–137. – URL https://www.ijcnis.org/index.php/ijcnis/article/viewFile/1375/168

[MacGillivray 2013] MacGillivray, Carrie: *The Internet of Things Is Poised to Change Everything Says IDC*. 2013. – URL http://www.businesswire.com/news/home/20131003005687/en/Internet-Things-Poised-Change-IDC. – eingesehen am: 26.01.2018

[Mahrenholz und Schuhmann 2017] Mahrenholz, Daniel ; Schuhmann, Ralf: Incident Response im SIEM Kontext - Ein Erfahrungsbericht. (2017)

[Manyika u. a. 2015] Manyika, James ; Chui, Michael ; Bisson, Peter ; Woetzel, Jonathan ; Dobbs, Richard ; Bughin, Jacques ; Aharon, Dan: THE INTERNET OF THINGS: MAPPING THE VALUE BEYOND THE HYPE. In: *Comput. Netw.* (2015). – URL https://www.mckinsey.de/files/unlocking_the_potential_of_the_internet_of_things_full_report.pdf

[Meffert u. a. 2017] Meffert, Christopher ; Clark, Devon ; Baggili, Ibrahim ; Breitinger, Frank: Forensic State Acquisition from Internet of Things (FSAIoT): A General Framework and Practical Approach for IoT Forensics Through IoT Device State Acquisition. In: *Proceedings of the 12th International Conference on Availability, Reliability and Security*. New York, NY, USA : ACM, 2017 (ARES '17), S. 56:1–56:11. – URL http://doi.acm.org/10.1145/3098954.3104053. – ISBN 978-1-4503-5257-4

[Meola 2016] Meola, Andrew: *Automotive Industry Trends: IoT Connected Smart Cars and Vehicles*. 2016. – URL http://www.businessinsider.de/internet-of-

things-connected-smart-cars-2016-10?r=US&IR=T. – eingesehen am: 26.01.2018

[Müller u. a. 2016] Müller, Christian ; Armknecht, Frederik ; Benenson, Zinaida ; Morgner, Philipp: On the security of the ZigBee light link touchlink commissioning procedure. In: Meier, Michael (Hrsg.) ; Reinhardt, Delphine (Hrsg.) ; Wendzel, Steffen (Hrsg.): *Sicherheit 2016 - Sicherheit, Schutz und Zuverlässigkeit.* Bonn : Gesellschaft für Informatik e.V., 2016, S. 229–240

[Notra u. a. 2014] Notra, S. ; Siddiqi, M. ; Gharakheili, H. H. ; Sivaraman, V. ; Boreli, R.: An experimental study of security and privacy risks with emerging household appliances. In: *2014 IEEE Conference on Communications and Network Security*, Oct 2014, S. 79–84

[O'Brien 2017] O'Brien, Dick: Ransomware 2017. In: *Internet Security Threat Report* (2017)

[Osborne 2015] Osborne, Charlie: Most companies take over six months to detect data breaches. (2015). – URL http://www.zdnet.com/article/businesses-take-over-six-months-to-detect-data-breaches

[Philips a] Philips: *hue Developers Program.* – URL https://developers.meethue.com/. – eingesehen: 08.12.2017

[Philips b] Philips: *Philips Hue.* – URL http://www2.meethue.com/de-de. – eingesehen: 14.12.2017

[proofpoint 2017] proofpoint: *Quarterly Threat Report Q1 2017.* 2017. – URL https://www.proofpoint.com/sites/default/files/pfpt-us-tr-q117-threat-report.pdf

[Puri und Dukatz 2015] Puri, C. ; Dukatz, C.: Analyzing and Predicting Security Event Anomalies: Lessons Learned from a Large Enterprise Big Data Streaming Analytics Deployment. In: *2015 26th International Workshop on Database and Expert Systems Applications (DEXA)*, Sept 2015, S. 152–158. – ISSN 1529-4188

[Reiber 2017] Reiber, Fabian: *Überblick diverser Plattformen zum Verarbeiten von Sicherheitsvorfällen und wie sie entsprechende Informationen untereinander austauschen können.* 2017

[retresco] retresco: *Die Giganten im Open Source Suchmark: Solr/Lucene vs. Elasticsearch - ein Vergleich.* – URL https://www.retresco.de/die-giganten-im-open-source-suchmarkt-solrlucene-vs-elasticsearch-ein-vergleich-wissensmanagement-magazin/. – eingesehen: 14.12.2017

[riverloopsec 2017] RIVERLOOPSEC: *ZBDump, zbwireshark etc problems.* 2017. – URL https://github.com/riverloopsec/killerbee/issues/86. – eingesehen am: 26.01.2018

[Romdhani u. a. 2015] ROMDHANI, Imed ; ABDMEZIEM, Riad ; TANDJAOUI, D: Architecting the Internet of Things: State of the Art. (2015), 07. ISBN ISSN: 2198-4182

[Ronen u. a. 2017] RONEN, E. ; SHAMIR, A. ; WEINGARTEN, A. O. ; O'FLYNN, C.: IoT Goes Nuclear: Creating a ZigBee Chain Reaction. In: *2017 IEEE Symposium on Security and Privacy (SP)*, May 2017, S. 195–212

[Rose und Ramsey 2016] ROSE, Anthony ; RAMSEY, Ben: Picking Bluetooth Low Energy Locks from a Quarter Mile Away. In: *Defcon 24* (2016)

[Sager 2014] SAGER, Tony: Killing Advanced Threats in Their Tracks: An Intelligent Approach to Attack Prevention. In: *SANS Institute Reading Room* (2014). – URL https://www.sans.org/reading-room/whitepapers/analyst/killing-advanced-threats-tracks-intelligent-approach-attack-prevention-35302

[Sethi und Sarangi 2017] SETHI, Pallavi ; SARANGI, Smruti R.: Internet of Things: Architectures, Protocols, and Applications. In: *Journal of Electrical and Computer Engineering* (2017), S. 25. – URL https://doi.org/10.1155/2017/9324035

[Sivaraman u. a. 2016] SIVARAMAN, Vijay ; CHAN, Dominic ; EARL, Dylan ; BORELI, Roksana: Smart-Phones Attacking Smart-Homes. In: *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. New York, NY, USA : ACM, 2016 (WiSec '16), S. 195–200. – URL http://doi.acm.org/10.1145/2939918.2939925. – ISBN 978-1-4503-4270-4

[Srinivas 2014] SRINIVAS, Babu V.: Security Operations Centre (SOC) in a Utility Organization. In: *SANS Institute Reading Room* (2014). – URL https://www.sans.org/reading-room/whitepapers/ICS/security-operations-centre-soc-utility-organization-35502

[Statista 2017] STATISTA: *Prognose zur Anzahl der vernetzten Geräte im Internet der Dinge (IoT) weltweit in den Jahren 2016 bis 2020 (in Millionen Einheiten).* 2017. – URL http://de.statista.com/statistik/daten/studie/537093/umfrage/anzahl-der-vernetzten-geraete-im-internet-der-dinge-iot-weltweit/. – eingesehen: 30.11.2017

[van der Stock u. a. 2017] Stock, Andrew van der ; Glas, Brian ; Smithline, Neil ; Gigler, Torsten: *OWASP Top 10 2017: The Ten Most Critical Web Application Security Risks*. 2017

[Suleman 2017] Suleman, Khidr: *Saubere Luft: Bosch und Intel tragen zum Kampf gegen Luftverschmutzung bei*. 2017. – URL https://www.intel.de/content/www/de/de/it-managers/bosch-connected-world-iot-air-pollution.html. – eingesehen am: 26.01.2018

[Tan u. a. 2014] Tan, Z. ; Nagar, U. T. ; He, X. ; Nanda, P. ; Liu, R. P. ; Wang, S. ; Hu, J.: Enhancing Big Data Security with Collaborative Intrusion Detection. In: *IEEE Cloud Computing* 1 (2014), Sept, Nr. 3, S. 27–33. – ISSN 2325-6095

[Thakar und Parekh 2016] Thakar, Bhavik ; Parekh, Chandresh: Advance Persistent Threat: Botnet. In: *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*. New York, NY, USA : ACM, 2016 (ICTCS '16), S. 143:1–143:6. – URL http://doi.acm.org/10.1145/2905055.2905360. – ISBN 978-1-4503-3962-9

[Torres 2015] Torres, Alissa: Building a World-Class Security Operations Center: A Roadmap. In: *SANS Institute Reading Room* (2015). – URL https://www.sans.org/reading-room/whitepapers/analyst/building-world-class-security-operations-center-roadmap-35907

[UKEssays 2013] UKEssays: Evolution Of Smart Homes. (2013), 11. – URL https://www.ukessays.com/dissertation/examples/computer-sciences/evolution-of-smart-homes.php. – eingesehen am: 26.01.2018

[Wendzel 2016] Wendzel, Steffen: How to Increase the Security of Smart Buildings? In: *Commun. ACM* 59 (2016), April, Nr. 5, S. 47–49. – URL http://doi.acm.org/10.1145/2828636. – ISSN 0001-0782

[Wright 2009] Wright, Joshua: KillerBee: Practical ZigBee Exploitation Framework. (2009). – URL http://www.willhackforsushi.com/presentations/toorcon11-wright.pdf. – eingesehen am: 26.01.2018

[Zegzhda u. a. 2016] Zegzhda, Peter ; Zegzhda, Dmitry ; Kalinin, Maxim ; Pechenkin, Alexander ; Minin, Alexander ; Lavrova, Daria: Safe Integration of SIEM Systems with Internet of Things: Data Aggregation, Integrity Control, and Bioinspired Safe Routing. In: *Proceedings of the 9th International Conference on Security of Information and Networks*.

New York, NY, USA : ACM, 2016 (SIN '16), S. 81–87. – URL http://doi.acm.org/10.1145/2947626.2947639. – ISBN 978-1-4503-4764-8

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 27.02.2018 Thomas Jäger